AWS CLOUD FORMATION

Deploying Web GIS Technology

ABSTRACT

The goal of this project is to use AWS Cloud Formation technology and the AWS Python SDK to develop the configuration files necessary to launch a cluster of GIS servers with ArcGIS Portal installed. This system will then be loaded with geospatial data and served out as web services for use in web mapping applications.

Charles Mateer

CSC E-90: Cloud Computing and the Internet of Things

Contents

Problem Statement

Configure and deploy a web geographic information systems (GIS) solution to Amazon Web Services (AWS) using the AWS Python SDK and AWS CloudFormation. The web GIS system will consist of an Esri ArcGIS Server, data store, web adapter, web server, and Portal for ArcGIS (web based admin software). Upon deployment, geospatial data will be loaded onto the web GIS and configured to output web mapping services for use in web mapping applications.

Description of Data

GIS data exists in a variety of formats and data models. At its core, GIS data is a digital representation of geographic phenomenon on the earth and consists of data regarding its georeferenced location on a globe (given a certain spatial reference system) and attribution data that describe that geographic feature (ex: feature type, length, area, elevation, population, and other feature specific qualities). The most common GIS data models are vector and raster; each with their own array of data formats (vector examples: shapefile, geodatabase, GeoJSON, CSV, etc.; raster examples: GeoTIFF, Esri GRID, ERDAS IMAGINE, MrSID, etc.). Given the wide array of GIS data formats, sophisticated database management software is generally required to work with, manage, and convert GIS data. Furthermore, to expose GIS data as web services for use in web maps and web based geospatial analysis, further sophisticated web servers (GIS servers and web adapters) are required to interface with the database and serve out GIS web services. This document will outline the process for installing highly available and elastic resources for hosting these GIS software stacks.

Public GIS data from the City of Richmond, VA will be used to populate the web GIS's data store (which will handle the data conversion and management process) with sample datasets to be used in web mapping applications. Data will include:

- Municipal boundaries
- Road centerlines
- Trails
- Hydrology
- Topography
- Vegetation

These data sets will be hosted as web services (by the GIS server) and used as basemap layers in example web maps (using the web server).

Description of Software

Traditionally, for small organizations, GIS data has been stored on LANs/WANs in network attached storage locations to be utilized by researchers and analysts within an organization. For larger organizations, and for organizations desiring to share GIS data outside of an organization, enterprise solutions have been necessary. Enterprise level GIS software stacks for hosting GIS web services and allowing interaction throughout a large organization generally include a GIS server, an enterprise level database, GIS integration software for the database, web servers, and web adapter software (for the GIS server and web server). This stack is often very complex to set up and can be cost prohibitive for smaller organizations as the licensing for these software packages is quite expensive.

One of the flagship companies providing GIS software stacks worldwide is Esri. In 2012, Esri released ArcGIS Online (AGOL) which is a cloud solution for web GIS. AGOL allows utilization of web GIS capabilities and resources hosted on Esri's servers and provides web based content management software for users to administer and manage their geospatial data. More recently, Esri has teamed up with AWS to provide AMIs with Esri software stacks and templates for AWS Cloud Formation. This allows users to stand-up their own web GIS clouds without having to rely on AGOL which has data size and usability restrictions.

The technology stack used in this guide will be outlined below.

AWS CloudFormation

AWS infrastructure as code service for standing up and managing software stacks across multiple AWS resources. Provides a centralized console for managing these resources and configuration templates that use best management practices for deployment and are easily tweaked. [Link]

Python

Python is an open source, dynamic programming language commonly used for scripting as it is easy to learn and read and provides powerful results with little coding overhead. We will be using version 2.7. [Link]

AWS Python SDK

The AWS SDK for Python is called Boto and is used for developing AWS applications and accessing AWS resources in Python. [Link]

ArcPy

ArcPy is a Python API for accessing Esri ArcGIS (described below) resources. [Link]

Esri Developer Network

Esri Developer Network (EDN) is a low cost (free student licenses may be available) software and license package that provides access to many Esri GIS technologies. It will have all the GIS software and licenses required for this guide (these GIS software stacks are described below). [Link]

Esri ArcGIS

Esri's entire catalog of GIS software and solutions for desktop, mobile, web, and IoT. [Link]

Esri ArcGIS for Desktop

Desktop GIS software used by end-users to work with, manage, and analyze GIS data. This software can be used to inspect and style (cartography) GIS data and will be used for publishing GIS web services to our web GIS. This version of ArcGIS also comes packaged with ArcPy and gives access to the tools necessary to publish GIS web services. [Link]

Esri ArcGIS for Server

Server-side technology for interfacing with a GIS database, managing user access/authentication, and responding to requests for GIS web services. [Link]

IIS Web Server
Web server for hosting web applications. This will be linked to the GIS Server via the web adapter and will route incoming traffic to the GIS server if GIS resources are requested. [Link]

Esri Web Adapter
Interface between the web server and the GIS server. [Link]

Esri Datastore
Specialized database technology that meshes with an existing RDBMS to allow storage of spatial data types. Will store our GIS data and web service definitions in the cloud when uploaded from a desktop environment. [Link]

Esri Portal for ArcGIS
Web-based content management system which provides an interface for users to interact with web GIS resources. [Link]

Esri ArcGIS JavaScript API
JavaScript API for creating web mapping applications. Able to consume web GIS services. We will use version 3.18. [Link]

Videos
Short Overview: https://www.youtube.com/watch?v=jNO5wFPRBO0&feature=youtu.be

Full Outline: https://www.youtube.com/watch?v=zd74Km1zFqA&feature=youtu.be

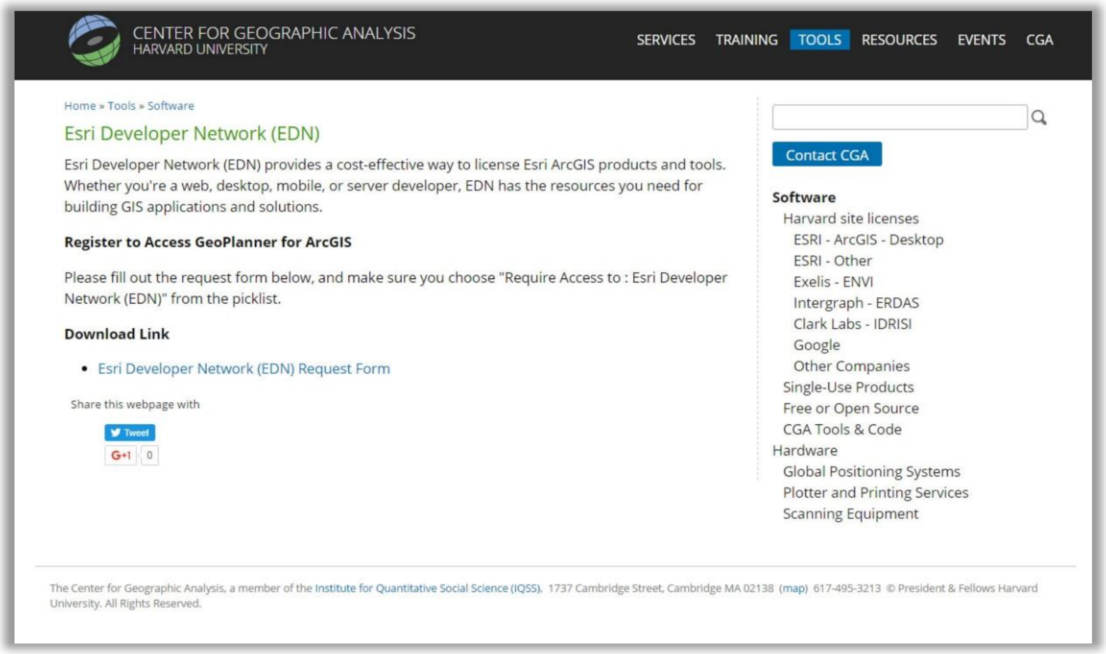
Live Demo: https://www.youtube.com/watch?v=PpL-0VdDCPs

Procedures
Obtain an EDN License and Install ArcGIS for Desktop
Obtain EDN License
Go to the website for the Harvard Center for Geographic Analysis (Tools Section):

http://gis.harvard.edu/tools

Select Esri – Other from the side panel on the right underneath "Software."

Go to the second page and select Esri Developer Network.


CENTER FOR GEOGRAPHIC ANALYSIS
HARVARD UNIVERSITY

SERVICES TRAINING TOOLS RESOURCES EVENTS CGA

Home » Tools » Software

Esri Developer Network (EDN)

Esri Developer Network (EDN) provides a cost-effective way to license Esri ArcGIS products and tools. Whether you're a web, desktop, mobile, or server developer, EDN has the resources you need for building GIS applications and solutions.

Register to Access GeoPlanner for ArcGIS

Please fill out the request form below, and make sure you choose "Require Access to : Esri Developer Network (EDN)" from the picklist.

Download Link

- Esri Developer Network (EDN) Request Form

Share this webpage with

Tweet
G+1 0

Contact CGA

Software
- Harvard site licenses
  - ESRI - ArcGIS - Desktop
  - ESRI - Other
  - Exelis - ENVI
  - Intergraph - ERDAS
  - Clark Labs - IDRISI
  - Google
  - Other Companies
- Single-Use Products
- Free or Open Source
- CGA Tools & Code

Hardware
- Global Positioning Systems
- Plotter and Printing Services
- Scanning Equipment

The Center for Geographic Analysis, a member of the Institute for Quantitative Social Science (IQSS), 1737 Cambridge Street, Cambridge MA 02138 (map) 617-495-3213 © President & Fellows Harvard University. All Rights Reserved.

Click the "Download Link," fill out the form, and submit.

You should get an e-mail shortly with the license.

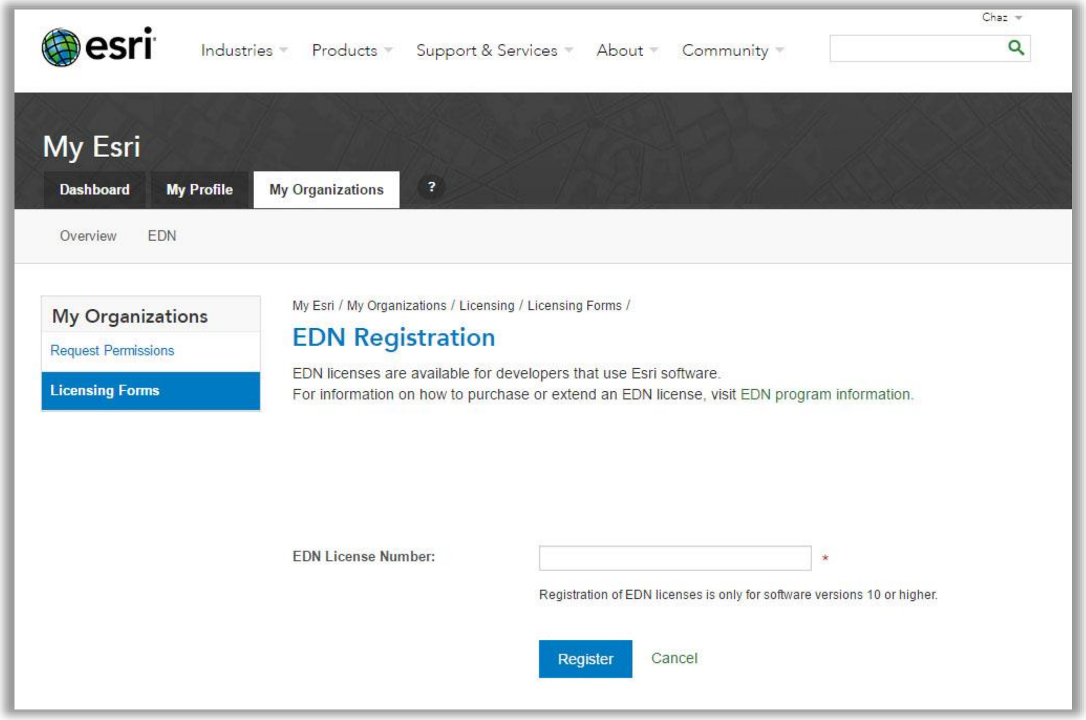
Create an ArcGIS Online Account

Go to the following link and sign up for an ArcGIS Online (AGOL) Developer's account (the developers account gives you some free tools that you would not get with the regular account – however, you likely won't end up needing them as the EDN license will cover most everything you need):

https://developers.arcgis.com/sign-up/

Now travel to the link that was sent to you from the Harvard Center for Geographic Analysis in regards to your EDN license and log-in with your new AGOL account.

Enter your EDN license and click "Register."



My Esri / My Organizations / Licensing / Licensing Forms /

EDN Registration

EDN licenses are available for developers that use Esri software.
For information on how to purchase or extend an EDN license, visit EDN program information.

EDN License Number: *

Registration of EDN licenses is only for software versions 10 or higher.

We will want to download two license files from this website for later use.

Travel to Overview > EDN > Authorizations (https://my.esri.com/#/edn/authorizations).

Select the following license:

- ArcGIS for Server Advanced Workgroup 10.1-10.4.1

Click on the Authorization number link.

"Create Provisioning File"

Check the box for "ArcGIS Server Advanced Workgroup" and under "Extensions" select "Portal for ArcGIS," then select "Next."

Fill out your personal information and either download or email the license to yourself.

Download and save both files for use later in this guide.

Install ArcGIS for Desktop (Optional)

You can either download and install ArcGIS for Desktop now, later, or never. It is not required to use AWS CloudFormation and Esri software. It will be necessary if you want to load the datasets we will be working with into your cloud though. ***The install is time consuming and requires requesting another license which I understand some people will not want to do and would rather get straight to working with the cloud technology.***

If you would like to install now; go back to the Harvard Center for Geographic Analysis Tools Section:

https://my.esri.com/#/edn/downloads

This time select Esri – ArcGIS – Desktop and select Desktop.

Under the "Download Link" section, select the ArcGIS Desktop 10.4.1 (or the most recent version) Software Package.

Also, select the Single-use License Request Form. Fill it out. Await an e-mail with your license.

Once you have downloaded ArcGIS Desktop, use the installation instructions, install wizard, and the single-use license from your e-mail to install ArcGIS Desktop.

ArcGIS Desktop will be used at the end of this guide to publish web services but is not a requirement.

Install Python and Required Libraries

Download and Install

Go to Python's download page and download your correct version of Python. Get at least version 3.0 or greater. Determine if you need 32 bit or 64 bit for your system. ***Note that if you installed ArcGIS Desktop, Python 2.7 will have been installed. We will ultimately use Python 2.7 for GIS data loading, and the AWS examples in this guide will be in Python 2.7. AWS has SDK version for both Python 2.7 and 3.5.***

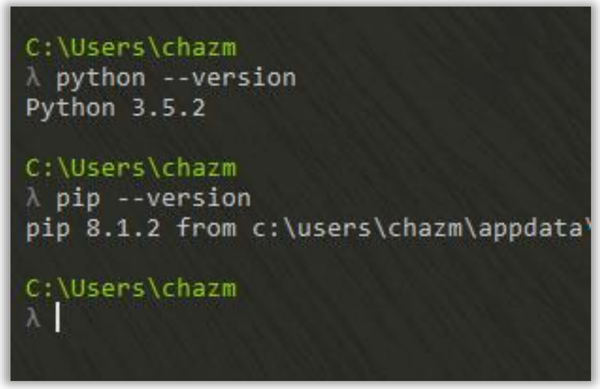
https://www.python.org/downloads/

Assuming you are running Windows; run the installer. If you see an option for adding Python to your path, be sure to select it. Otherwise, after the installation you will need to add the location of the Python executable to your PATH environment variable.

Once installed, open a command prompt, and verify that both Python and pip are present. (note that you will want to make sure you have version 2.7.x).

```
C: > python --version

C: > pip –version
```



```
C:\Users\chazm
λ python --version
Python 3.5.2

C:\Users\chazm
λ pip --version
pip 8.1.2 from c:\users\chazm\appdata\

C:\Users\chazm
λ
```

Make sure the version numbers are output. Otherwise, revisit the documentation and make sure that Python is in your PATH.

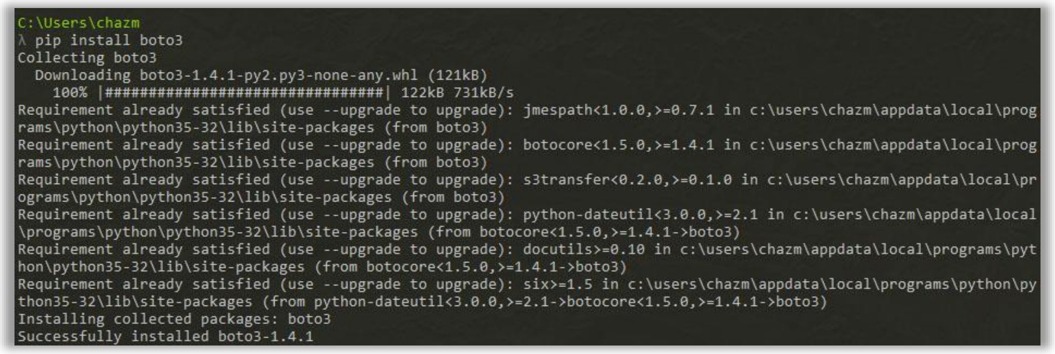
Install Boto

As described in the AWS SDK for Python (Boto – old version) documentation:

https://aws.amazon.com/sdk-for-python/

You can install Boto (or Boto3, though we will use Boto) using pip:

```
C: > pip install boto
```

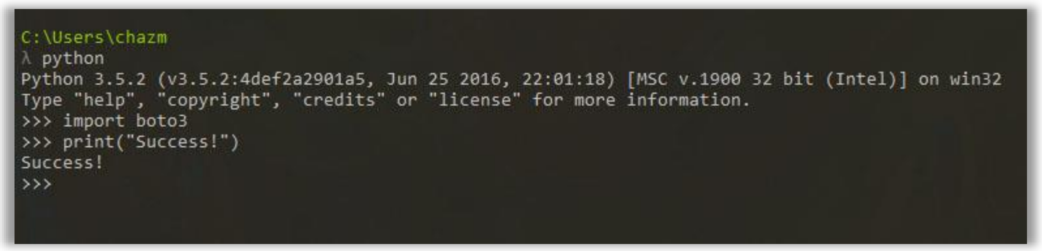


```
C:\Users\chazm
λ pip install boto3
Collecting boto3
  Downloading boto3-1.4.1-py2.py3-none-any.whl (121kB)
    100% |################################| 122kB 731kB/s
Requirement already satisfied (use --upgrade to upgrade): jmespath<1.0.0,>=0.7.1 in c:\users\chazm\appdata\local\prog
rams\python\python35-32\lib\site-packages (from boto3)
Requirement already satisfied (use --upgrade to upgrade): botocore<1.5.0,>=1.4.1 in c:\users\chazm\appdata\local\prog
rams\python\python35-32\lib\site-packages (from boto3)
Requirement already satisfied (use --upgrade to upgrade): s3transfer<0.2.0,>=0.1.0 in c:\users\chazm\appdata\local\pr
ograms\python\python35-32\lib\site-packages (from boto3)
Requirement already satisfied (use --upgrade to upgrade): python-dateutil<3.0.0,>=2.1 in c:\users\chazm\appdata\local
\programs\python\python35-32\lib\site-packages (from botocore<1.5.0,>=1.4.1->boto3)
Requirement already satisfied (use --upgrade to upgrade): docutils>=0.10 in c:\users\chazm\appdata\local\programs\pyt
hon\python35-32\lib\site-packages (from botocore<1.5.0,>=1.4.1->boto3)
Requirement already satisfied (use --upgrade to upgrade): six>=1.5 in c:\users\chazm\appdata\local\programs\python\py
thon35-32\lib\site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.5.0,>=1.4.1->boto3)
Installing collected packages: boto3
Successfully installed boto3-1.4.1
```

If you previously installed AWS CLI Tools then your AWS Credentials should already be set up. Otherwise, follow the directions here to get them set up:

https://boto3.readthedocs.io/en/latest/guide/quickstart.html

Verify that Boto was successfully installed by opening the Python REPL and importing Boto (if it fails to import it was not installed correctly).



```
C:\Users\chazm
λ python
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import boto3
>>> print("Success!")
Success!
>>>
```

Use AWS CloudFormation to Deploy a Web GIS

Accept the Esri Terms of Service

Before using Esri AMIs you must first accept the terms of service for an Esri AMI. Make sure you are logged into your AWS account and travel to the following:

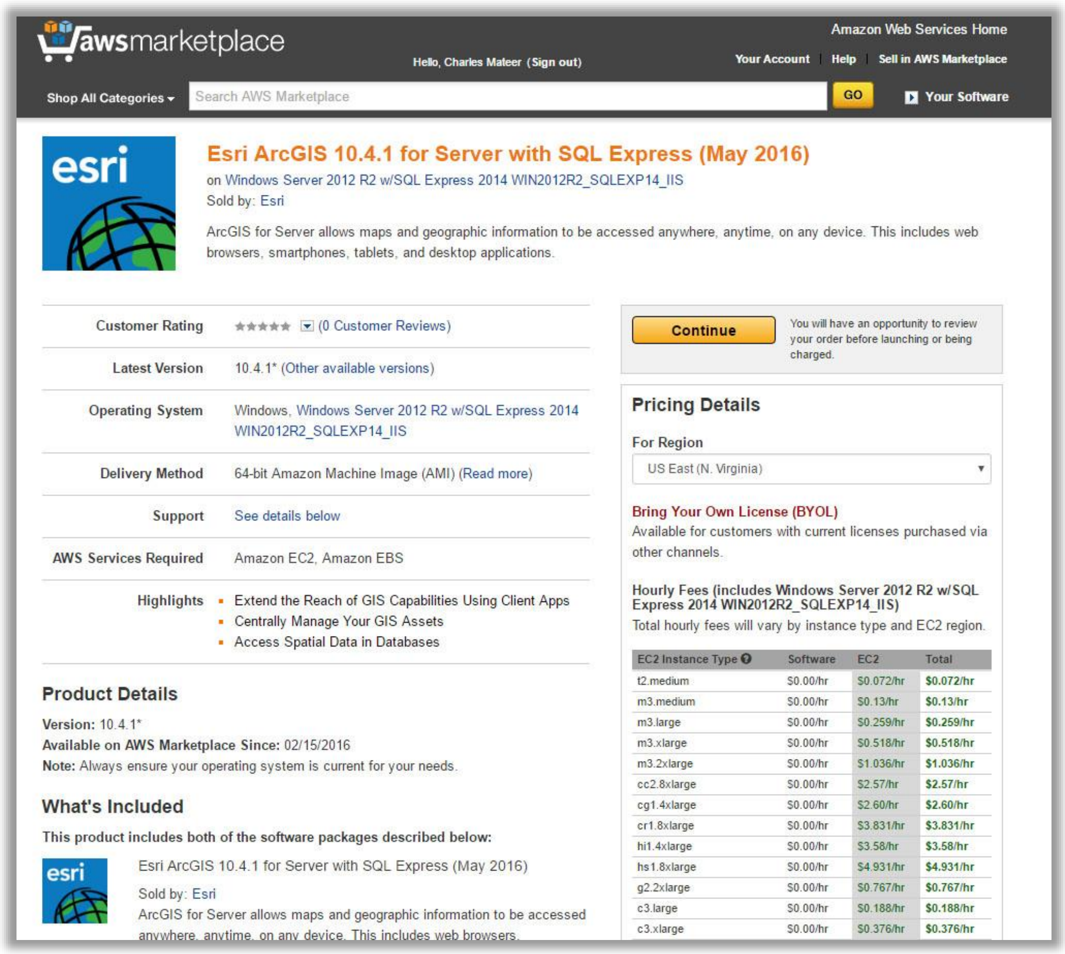
List of all terms of service:

http://server.arcgis.com/en/server/latest/cloud/amazon/aws-marketplace-accept-terms.htm

The ToS for the AMI I will be using (Esri ArcGIS 10.4.1 for Server with SQL Server Express):

https://aws.amazon.com/marketplace/pp/B01BWL2WZQ

On this page, you will see various information about the AMI such as the operating system, software that comes pre-installed, AWS resources that will be used, and a cost estimator. You will find the ToS at the bottom of the page.



Esri ArcGIS 10.4.1 for Server with SQL Express (May 2016)
on Windows Server 2012 R2 w/SQL Express 2014 WIN2012R2_SQLEXP14_IIS
Sold by: Esri

ArcGIS for Server allows maps and geographic information to be accessed anywhere, anytime, on any device. This includes web browsers, smartphones, tablets, and desktop applications.

Customer Rating	★★★★★ (0 Customer Reviews)
Latest Version	10.4.1* (Other available versions)
Operating System	Windows, Windows Server 2012 R2 w/SQL Express 2014 WIN2012R2_SQLEXP14_IIS
Delivery Method	64-bit Amazon Machine Image (AMI) (Read more)
Support	See details below
AWS Services Required	Amazon EC2, Amazon EBS
Highlights	• Extend the Reach of GIS Capabilities Using Client Apps • Centrally Manage Your GIS Assets • Access Spatial Data in Databases

You will have an opportunity to review your order before launching or being charged.

Pricing Details

For Region
US East (N. Virginia)

Bring Your Own License (BYOL)
Available for customers with current licenses purchased via other channels.

Hourly Fees (includes Windows Server 2012 R2 w/SQL Express 2014 WIN2012R2_SQLEXP14_IIS)
Total hourly fees will vary by instance type and EC2 region.

EC2 Instance Type	Software	EC2	Total
t2.medium	$0.00/hr	$0.072/hr	$0.072/hr
m3.medium	$0.00/hr	$0.13/hr	$0.13/hr
m3.large	$0.00/hr	$0.259/hr	$0.259/hr
m3.xlarge	$0.00/hr	$0.518/hr	$0.518/hr
m3.2xlarge	$0.00/hr	$1.036/hr	$1.036/hr
cc2.8xlarge	$0.00/hr	$2.57/hr	$2.57/hr
cg1.4xlarge	$0.00/hr	$2.60/hr	$2.60/hr
cr1.8xlarge	$0.00/hr	$3.831/hr	$3.831/hr
hi1.4xlarge	$0.00/hr	$3.58/hr	$3.58/hr
hs1.8xlarge	$0.00/hr	$4.931/hr	$4.931/hr
g2.2xlarge	$0.00/hr	$0.767/hr	$0.767/hr
c3.large	$0.00/hr	$0.188/hr	$0.188/hr
c3.xlarge	$0.00/hr	$0.376/hr	$0.376/hr

Product Details

Version: 10.4.1*
Available on AWS Marketplace Since: 02/15/2016
Note: Always ensure your operating system is current for your needs.

What's Included

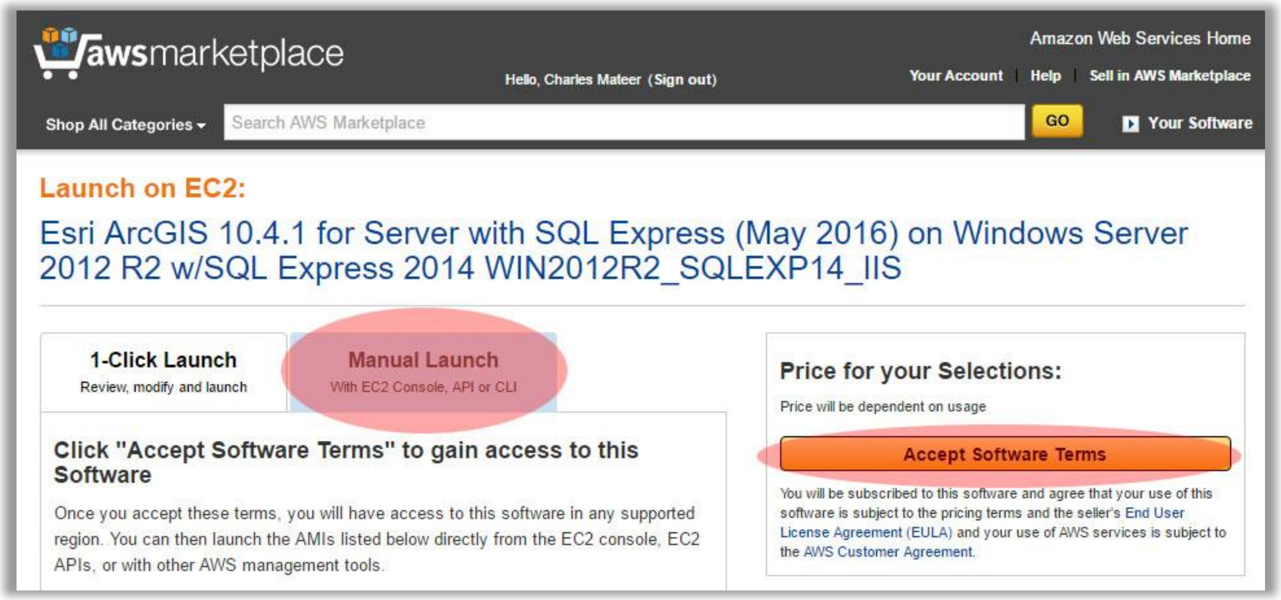
This product includes both of the software packages described below:

Esri ArcGIS 10.4.1 for Server with SQL Express (May 2016)
Sold by: Esri
ArcGIS for Server allows maps and geographic information to be accessed anywhere, anytime, on any device. This includes web browsers,

Under "For Region," select your region.

Select "Continue".

Switch to the "Manual Launch" tab and select "Accept Software Terms."



Launch on EC2:

Esri ArcGIS 10.4.1 for Server with SQL Express (May 2016) on Windows Server 2012 R2 w/SQL Express 2014 WIN2012R2_SQLEXP14_IIS

1-Click Launch
Review, modify and launch

Manual Launch
With EC2 Console, API or CLI

Click "Accept Software Terms" to gain access to this Software

Once you accept these terms, you will have access to this software in any supported region. You can then launch the AMIs listed below directly from the EC2 console, EC2 APIs, or with other AWS management tools.

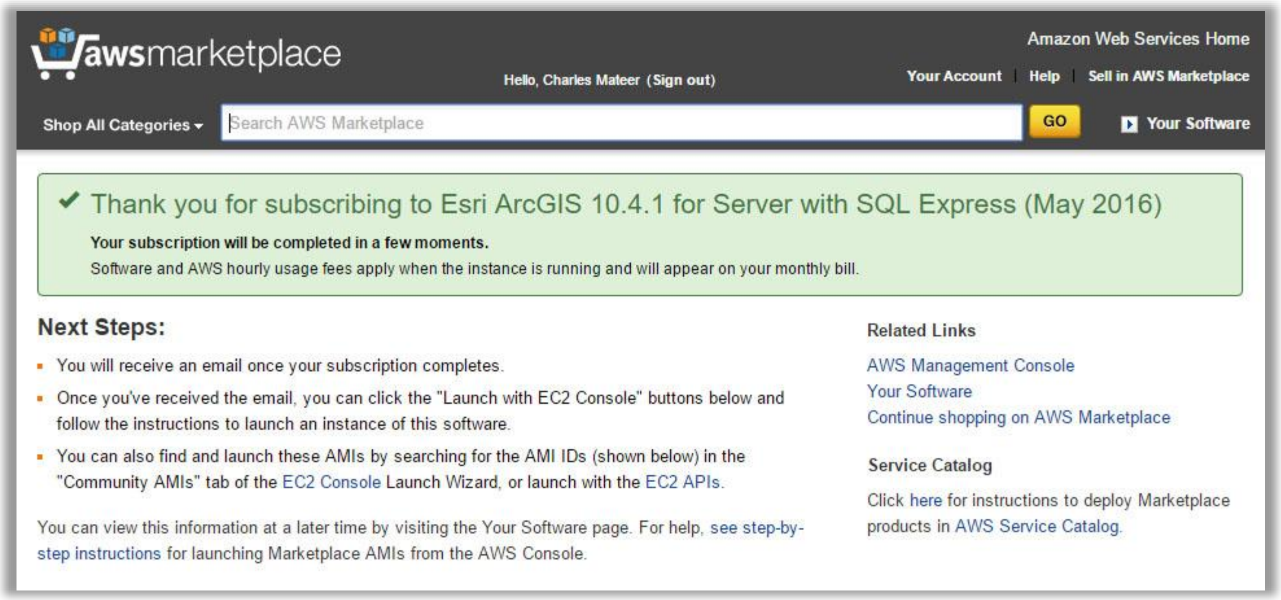
Price for your Selections:

Price will be dependent on usage

Accept Software Terms

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 (EULA) and your use of AWS services is subject to the AWS Customer Agreement.

You should get a success message.



✔ Thank you for subscribing to Esri ArcGIS 10.4.1 for Server with SQL Express (May 2016)

Your subscription will be completed in a few moments.
Software and AWS hourly usage fees apply when the instance is running and will appear on your monthly bill.

Next Steps:

- You will receive an email once your subscription completes.
- Once you've received the email, you can click the "Launch with EC2 Console" buttons below and follow the instructions to launch an instance of this software.
- You can also find and launch these AMIs by searching for the AMI IDs (shown below) in the "Community AMIs" tab of the EC2 Console Launch Wizard, or launch with the EC2 APIs.

You can view this information at a later time by visiting the Your Software page. For help, see step-by-step instructions for launching Marketplace AMIs from the AWS Console.

Related Links

AWS Management Console
Your Software
Continue shopping on AWS Marketplace

Service Catalog

Click here for instructions to deploy Marketplace products in AWS Service Catalog.

You should also get a confirmation email that specifies that you will now be able to access the Esri ArcGIS AMI in the AWS Console and/or through APIs.

Determine a Web GIS Stack to Use

You can use AWS CloudFormation to deploy any Web GIS architecture you would like but they do provide a few templates to work off based on commonly used architectures:

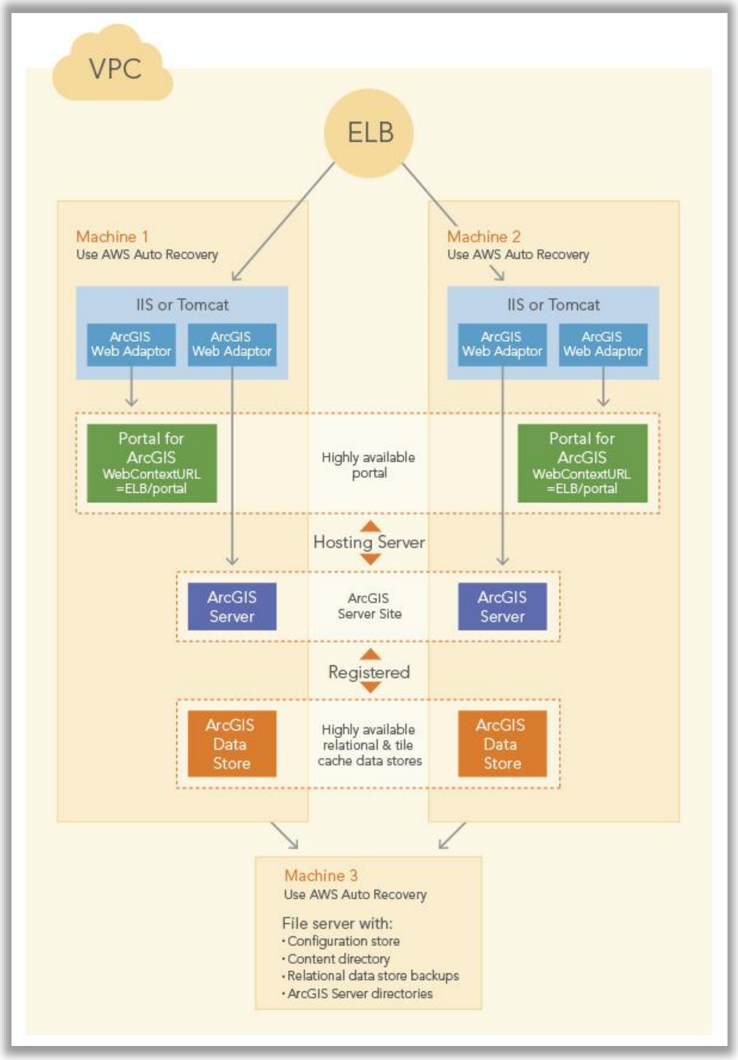
Highly available single-machine GIS server site – Sets up an ELB and several EC2 instances with ArcGIS Server software under one VPC.

Highly available GIS server site – Sets up a webserver, web adaptor, data server, and one or more ArcGIS server machines controlled by an ELB.

Web GIS on one machine – All necessary software for a web GIS on one EC2 instance.

Highly available web GIS – Same as the web GIS but cloned across multiple machines on an ELB with an additional machine to store configuration files.

We will use the highly available web GIS for this project. A diagram of this is shown below and further research can be gathered at this location: http://server.arcgis.com/en/server/latest/cloud/amazon/aws-cloud-formation-and-arcgis-server.htm#ESRI_SECTION1_C5F91B1AE98E452093CEC65BF528E9B9



AWS CloudFormation will use JSON configuration templates to deploy our web GIS which is what we will download next.

Obtain Configuration Template and Prepare

Esri AWS CloudFormation templates can be found here:
http://arcgisstore1041.s3.amazonaws.com/5686/docs/index.html

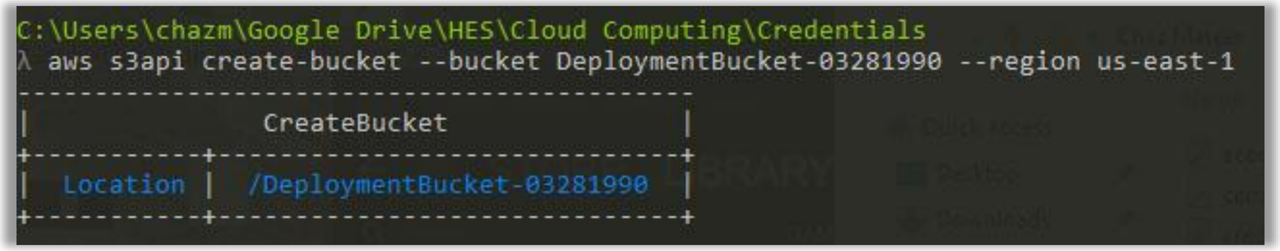
Templates can be downloaded by selecting “View”, researched by selecting “Read Me”, or launched through AWS Console by selecting “Launch Stack.”

We will be using the AWS Python API and the JSON template configurations to automate the construction of our web GIS rather than AWS console. First though we need to set up a few things on our AWS account.

1) Create an S3 bucket called “DeploymentBucket-{uid}” (you can also do this step via AWS Console):

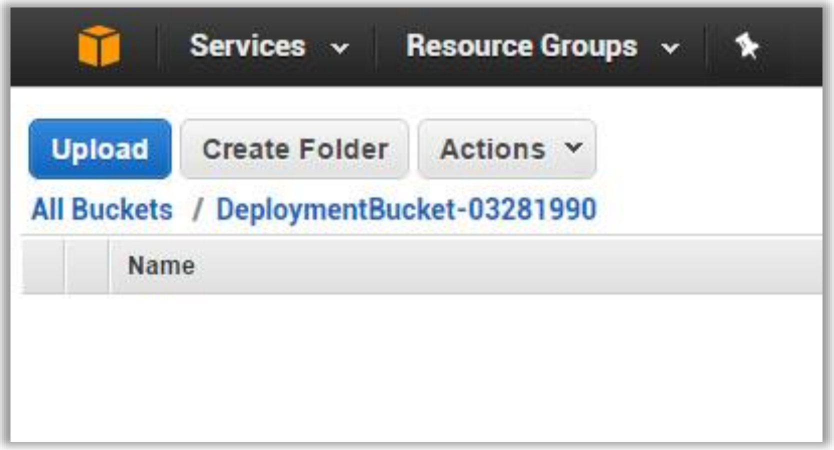
```
C: > aws s3api create-bucket --bucket DeploymentBucket-03281990 --
region us-east-1
```

Output:



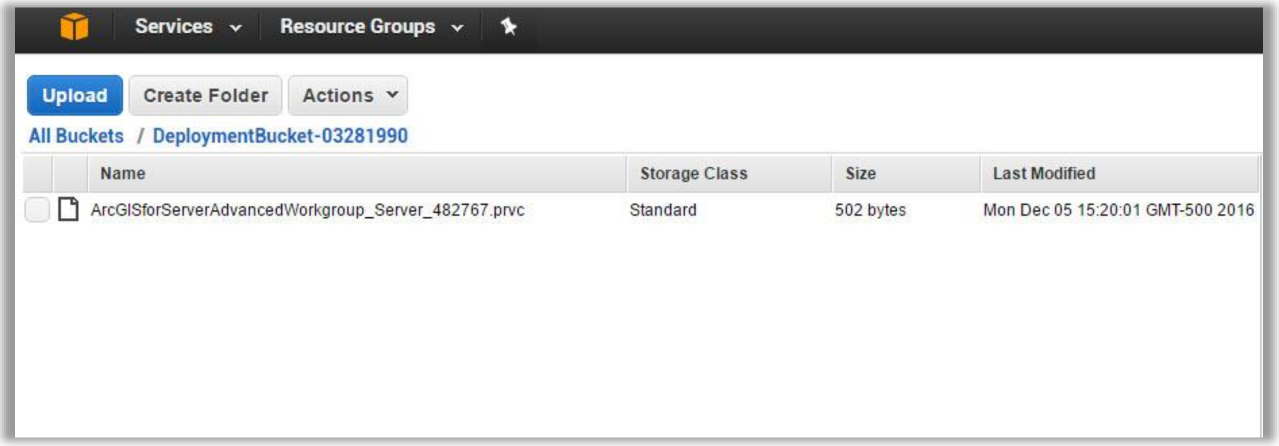
```
C:\Users\chazm\Google Drive\HES\Cloud Computing\Credentials
λ aws s3api create-bucket --bucket DeploymentBucket-03281990 --region us-east-1
-------------------------------------------
|              CreateBucket               |
+-----------+-----------------------------+
|  Location |  /DeploymentBucket-03281990 |
+-----------+-----------------------------+
```

View in AWS Console to verify:



Upload your ArcGIS Portal and ArcGIS Server license to this S3 bucket.

Additionally, you could also upload an SSL certificate to this S3 bucket if you were planning to use this site in a production environment with HTTPS.



All Buckets / DeploymentBucket-03281990

Name	Storage Class	Size	Last Modified
ArcGISforServerAdvancedWorkgroup_Server_482767.prvc	Standard	502 bytes	Mon Dec 05 15:20:01 GMT-500 2016

Now download a copy of the following template: "High-available Web GIS in EC2-VPC (Windows). By clicking "View."

Save the resulting JSON file.

Prepare Python Script for Automated Deploying

Make sure Python and boto3 are installed as specified above in the "Install Python and Required Libraries" section.

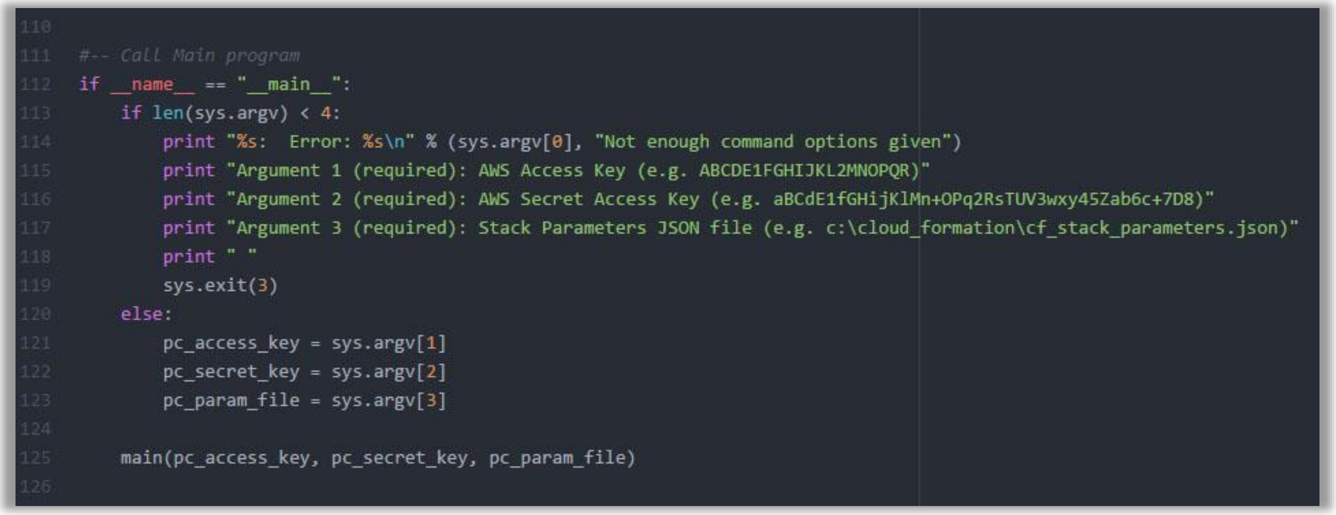
Download and unzip the Python templates from:
http://arcgisstore1041.s3.amazonaws.com/5686/tools/python.zip

We will be using `cloudformation_stack_creation_104.py` and `cf_parameters_win_HA_webgisstack.json` to launch our stack.

Review of the `cloudformation_stack_creation_104.py`

We will run this file from the command line using specific parameters.

If we go to line 111 we can see the main function call:



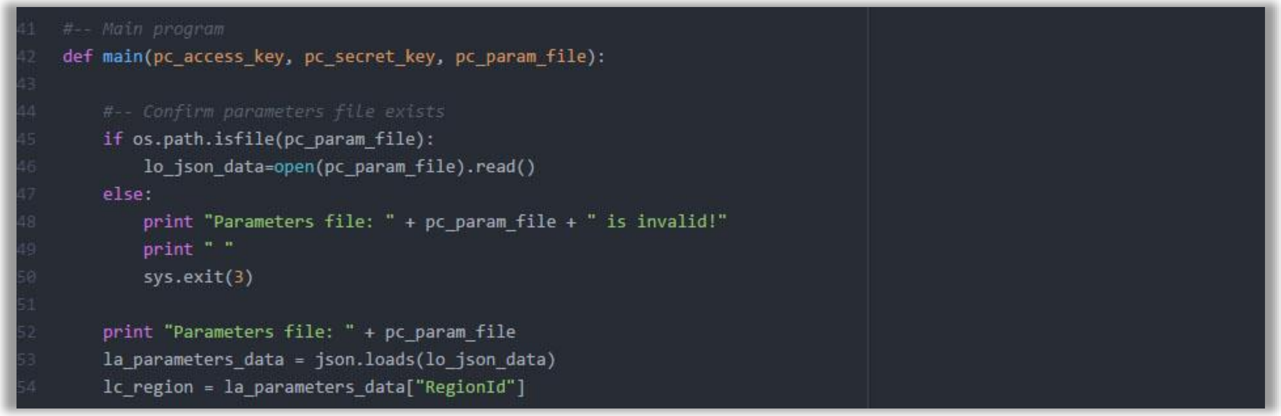
```
110
111  #-- Call Main program
112  if __name__ == "__main__":
113      if len(sys.argv) < 4:
114          print "%s:  Error: %s\n" % (sys.argv[0], "Not enough command options given")
115          print "Argument 1 (required): AWS Access Key (e.g. ABCDE1FGHIJKL2MNOPQR)"
116          print "Argument 2 (required): AWS Secret Access Key (e.g. aBCdE1fGHijKlMn+OPq2RsTUV3wxy45Zab6c+7D8)"
117          print "Argument 3 (required): Stack Parameters JSON file (e.g. c:\cloud_formation\cf_stack_parameters.json)"
118          print " "
119          sys.exit(3)
120      else:
121          pc_access_key = sys.argv[1]
122          pc_secret_key = sys.argv[2]
123          pc_param_file = sys.argv[3]
124
125      main(pc_access_key, pc_secret_key, pc_param_file)
126
```

An if block is used to do some error handling in case no arguments are passed via the command line.

Then, the following arguments are saved to variables and passed to the main function:

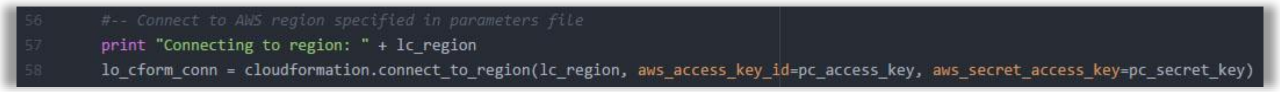
- pc_access_key = the user's AWS access key
- pc_secret_key = the user's AWS secret key
- pc_param_file = our parameters file (`cf_parameters_win_HA_webgisstack.json`)

The main function starts on line 42 with a check to make sure the input parameters are valid.



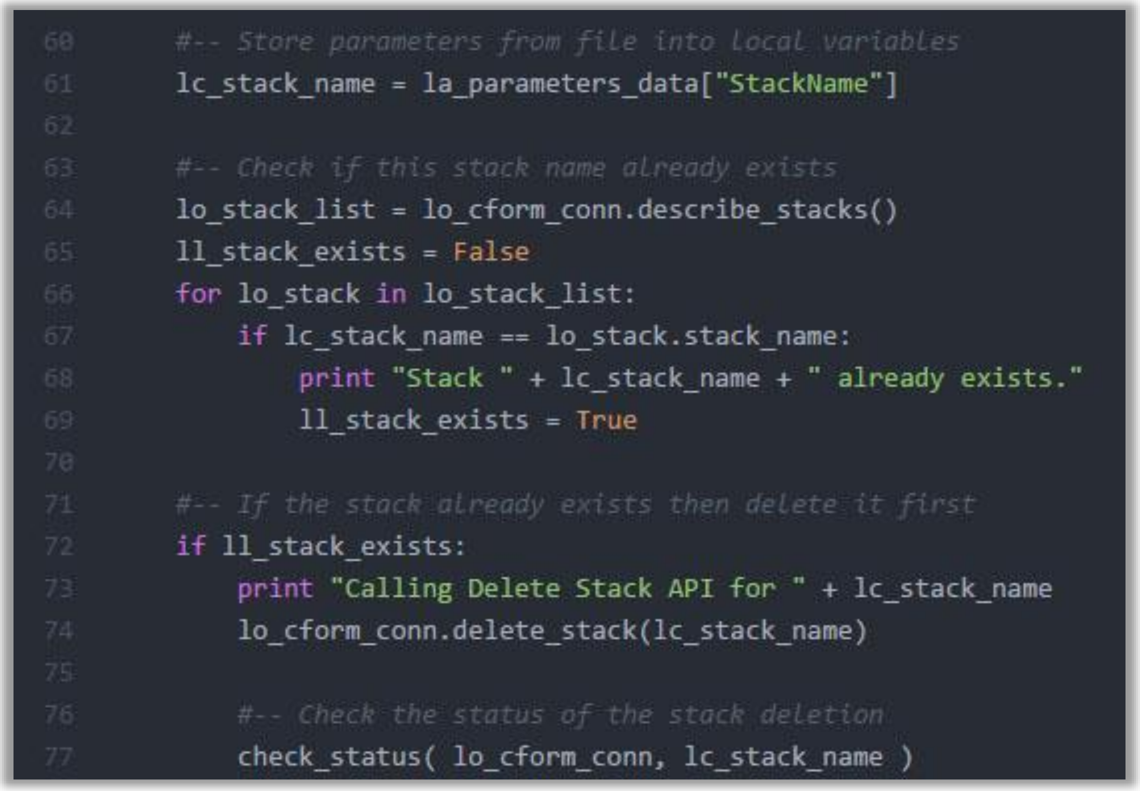
```
41  #-- Main program
42  def main(pc_access_key, pc_secret_key, pc_param_file):
43
44      #-- Confirm parameters file exists
45      if os.path.isfile(pc_param_file):
46          lo_json_data=open(pc_param_file).read()
47      else:
48          print "Parameters file: " + pc_param_file + " is invalid!"
49          print " "
50          sys.exit(3)
51
52      print "Parameters file: " + pc_param_file
53      la_parameters_data = json.loads(lo_json_data)
54      lc_region = la_parameters_data["RegionId"]
```

Line 58: Create a CloudFormation connection object using AWS credentials and region.



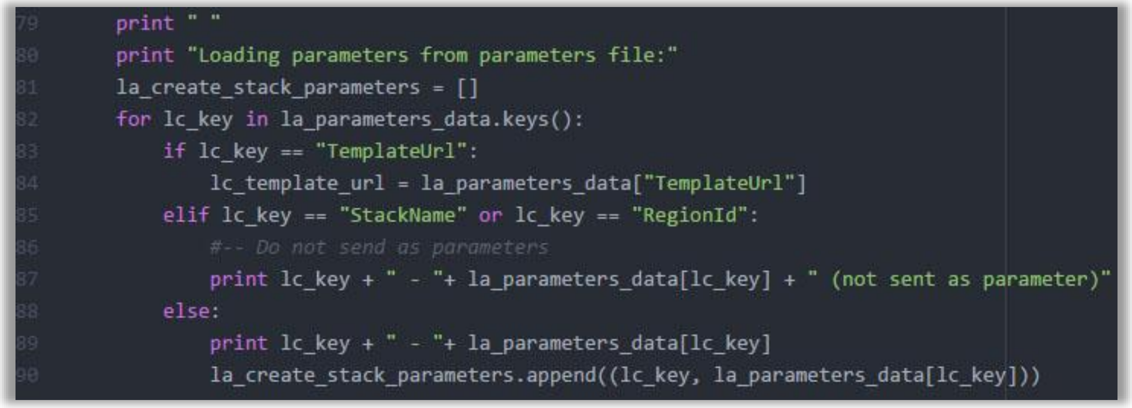
```
56      #-- Connect to AWS region specified in parameters file
57      print "Connecting to region: " + lc_region
58      lo_cform_conn = cloudformation.connect_to_region(lc_region, aws_access_key_id=pc_access_key, aws_secret_access_key=pc_secret_key)
```

Line 61-77: Set the software stack name and check the AWS account to see if it exists (deleting if so).



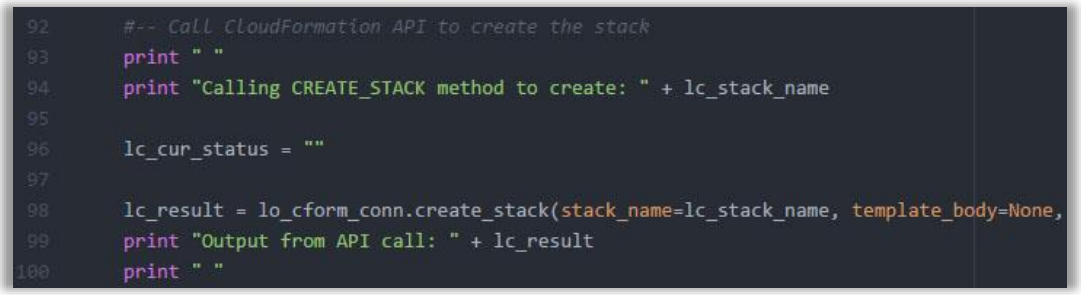
```
60      #-- Store parameters from file into local variables
61      lc_stack_name = la_parameters_data["StackName"]
62
63      #-- Check if this stack name already exists
64      lo_stack_list = lo_cform_conn.describe_stacks()
65      ll_stack_exists = False
66      for lo_stack in lo_stack_list:
67          if lc_stack_name == lo_stack.stack_name:
68              print "Stack " + lc_stack_name + " already exists."
69              ll_stack_exists = True
70
71      #-- If the stack already exists then delete it first
72      if ll_stack_exists:
73          print "Calling Delete Stack API for " + lc_stack_name
74          lo_cform_conn.delete_stack(lc_stack_name)
75
76          #-- Check the status of the stack deletion
77          check_status( lo_cform_conn, lc_stack_name )
```

Line 81-90: Unload key values from the web GIS software stack into variables for use later in the script.



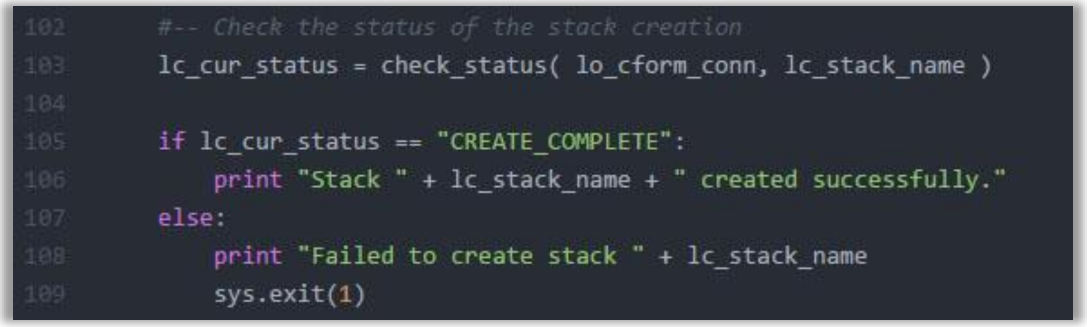
```
79      print " "
80      print "Loading parameters from parameters file:"
81      la_create_stack_parameters = []
82      for lc_key in la_parameters_data.keys():
83          if lc_key == "TemplateUrl":
84              lc_template_url = la_parameters_data["TemplateUrl"]
85          elif lc_key == "StackName" or lc_key == "RegionId":
86              #-- Do not send as parameters
87              print lc_key + " - "+ la_parameters_data[lc_key] + " (not sent as parameter)"
88          else:
89              print lc_key + " - "+ la_parameters_data[lc_key]
90              la_create_stack_parameters.append((lc_key, la_parameters_data[lc_key]))
```

Line 98: Use the CloudFormation connection to create the stack suing the parameters from the configuration file.



```
92      #-- Call CloudFormation API to create the stack
93      print " "
94      print "Calling CREATE_STACK method to create: " + lc_stack_name
95
96      lc_cur_status = ""
97
98      lc_result = lo_cform_conn.create_stack(stack_name=lc_stack_name, template_body=None,
99      print "Output from API call: " + lc_result
100     print " "
```

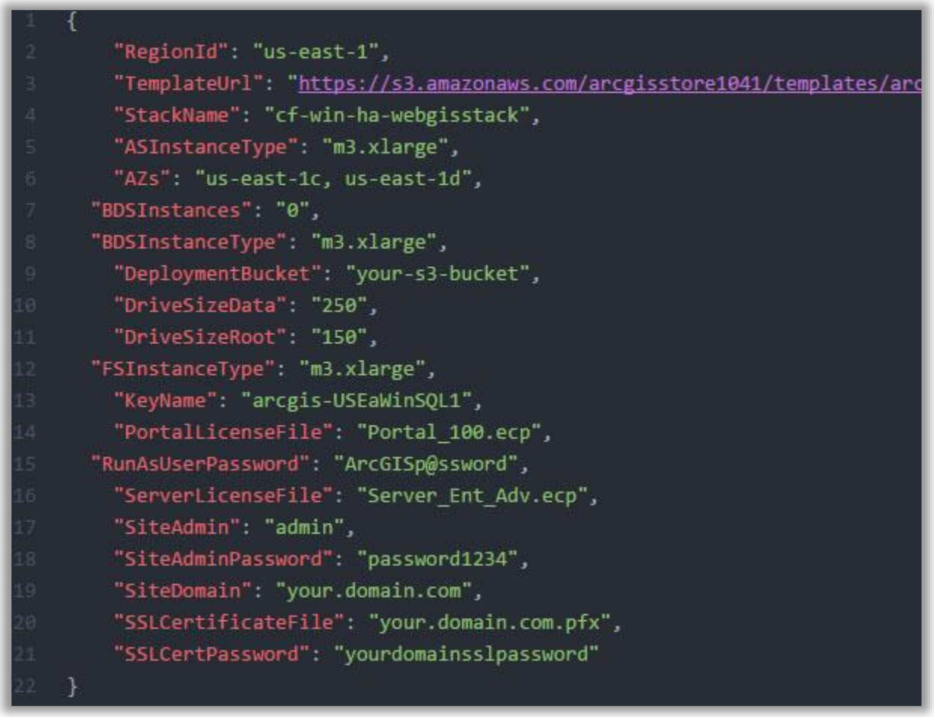
Line 103-109: Monitor the creation and exit if necessary (if it fails at runtime). Uses logic from the check_status function on lines 10-37.



```
102     #-- Check the status of the stack creation
103     lc_cur_status = check_status( lo_cform_conn, lc_stack_name )
104
105     if lc_cur_status == "CREATE_COMPLETE":
106         print "Stack " + lc_stack_name + " created successfully."
107     else:
108         print "Failed to create stack " + lc_stack_name
109         sys.exit(1)
```

Review the Configuration File

The configuration file specifies various specifications for your stack:



```
{
    "RegionId": "us-east-1",
    "TemplateUrl": "https://s3.amazonaws.com/arcgisstore1041/templates/arc
    "StackName": "cf-win-ha-webgisstack",
    "ASInstanceType": "m3.xlarge",
    "AZs": "us-east-1c, us-east-1d",
  "BDSInstances": "0",
  "BDSInstanceType": "m3.xlarge",
    "DeploymentBucket": "your-s3-bucket",
    "DriveSizeData": "250",
    "DriveSizeRoot": "150",
  "FSInstanceType": "m3.xlarge",
    "KeyName": "arcgis-USEaWinSQL1",
    "PortalLicenseFile": "Portal_100.ecp",
  "RunAsUserPassword": "ArcGISp@ssword",
    "ServerLicenseFile": "Server_Ent_Adv.ecp",
    "SiteAdmin": "admin",
    "SiteAdminPassword": "password1234",
    "SiteDomain": "your.domain.com",
    "SSLCertificateFile": "your.domain.com.pfx",
    "SSLCertPassword": "yourdomainsslpassword"
}
```

- **RegionId** – AWS region. We will be using us-east-1.

- **TemplateUrl –** Link to the JSON template for creating a highly available Web GIS stack via CloudFormation (described in the section below).
- **StackName** – A name for the stack. We will change this to CSC90-web-gis-final.
- **ASInstanceType** – Instance type. We will use m3.large.
- **AZs –** Availability zones.
- **BDSInstances –** Big data store. We will not be using this and it will be set to 0.
- **BDSInstanceType –** BDS instance type.
- **DeploymentBucket –** The name of the S3 bucket we made above.
- **DriveSizeData -** The size of the data drive in GB.
- **DriveSizeRoot** - The size of the root drive in GB.
- **FSInstanceType –** Instance type for file system machine. We will use m3.large.
- **KeyName –** Name of key pair for remote access.
- **PortalLicenseFile –** Name of the ArcGIS Portal license file.
- **RunAsUserPassword –** User password.
- **ServerLicenseFile -** Name of the ArcGIS Server license file.
- **SiteAdmin –** Site admin username.
- **SiteAdminPassword –** Site admin password.
- **SiteDomain –** Domain. I chose to purchase a cheap domain using Route 53 and set up an elastic IP for it. This worked out quite nicely.
- **SSLCertificateFile-** Name of SSL certificate file. Rather than purchase a certificate (since this is just a demo), I made a self-signed certificate using openssl and then converted to a .pfx file.
- **SSLCertPassword –** SSL Certificate password.

Review the Web GIS Stack Template

The Web GIS template is pulled from an Esri S3 bucket linked in the above-mentioned configuration file. It is quite long (1,208 lines of JSON) and describes links to specific resources (ex. AMIs) that CloudFormation will need to stand-up a web GIS architecture.

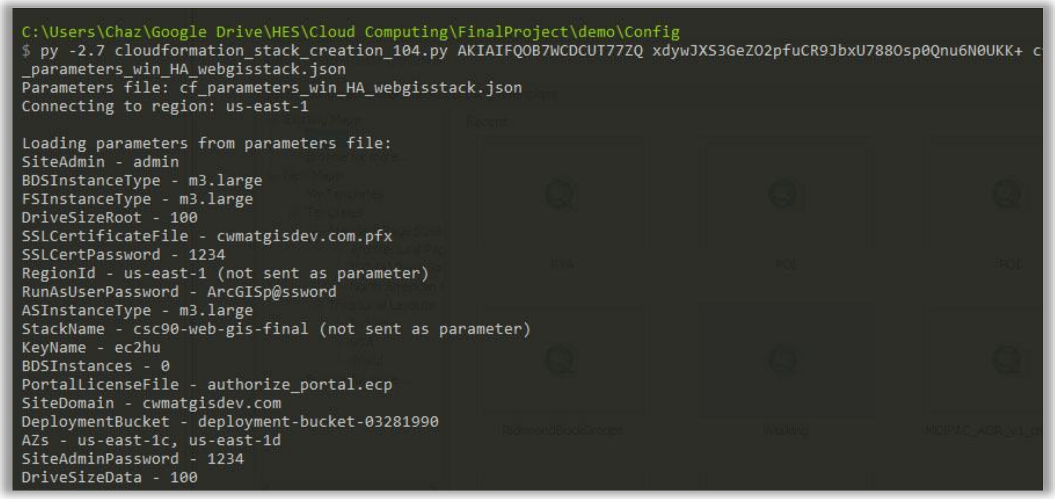
We will not need to update this file as our alterations to the configuration file (above) will make the changes we need.

Run the CloudFormation Python Script

Open a command line and issue the following command:

```
C: > python cloudformation_stack_creation_104.py {AWS Access Key} {AWS Secret Key}
cf_parameters_win_HA_webgisstack.json
```

We get the following command line output which gives us status updates on the launch progress.



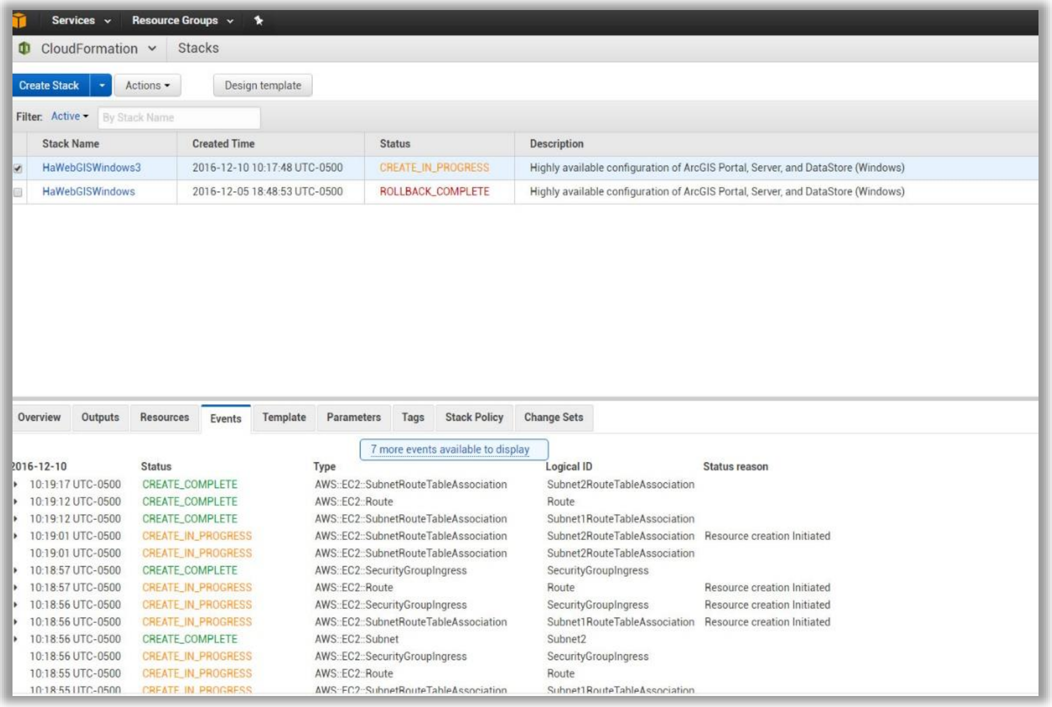
```
C:\Users\Chaz\Google Drive\HES\Cloud Computing\FinalProject\demo\Config
$ py -2.7 cloudformation_stack_creation_104.py AKIAIFQOB7WCDCUT77ZQ xdywJXS3GeZO2pfuCR9JbxU788Osp0Qnu6N0UKK+ c
_parameters_win_HA_webgisstack.json
Parameters file: cf_parameters_win_HA_webgisstack.json
Connecting to region: us-east-1

Loading parameters from parameters file:
SiteAdmin - admin
BDSInstanceType - m3.large
FSInstanceType - m3.large
DriveSizeRoot - 100
SSLCertificateFile - cwmatgisdev.com.pfx
SSLCertPassword - 1234
RegionId - us-east-1 (not sent as parameter)
RunAsUserPassword - ArcGISp@ssword
ASInstanceType - m3.large
StackName - csc90-web-gis-final (not sent as parameter)
KeyName - ec2hu
BDSInstances - 0
PortalLicenseFile - authorize_portal.ecp
SiteDomain - cwmatgisdev.com
DeploymentBucket - deployment-bucket-03281990
AZs - us-east-1c, us-east-1d
SiteAdminPassword - 1234
DriveSizeData - 100
```

Additionally, we can go to the AWS console to monitor the CloudFormation logs.

AWS > Services > CloudFormation > Then select your recently created stack.

The "Events" tab at the bottom gives you useful logs on what is happening as your stack is stood up. You can additionally see these and more in-depth logs via Cloud Watch > Logs. In the image below you will see an event log as my stack is being stood up (note also that I have a failed attempt still in my viewport - you will see that failed attempts are rolled back so that resources aren't being wasted).

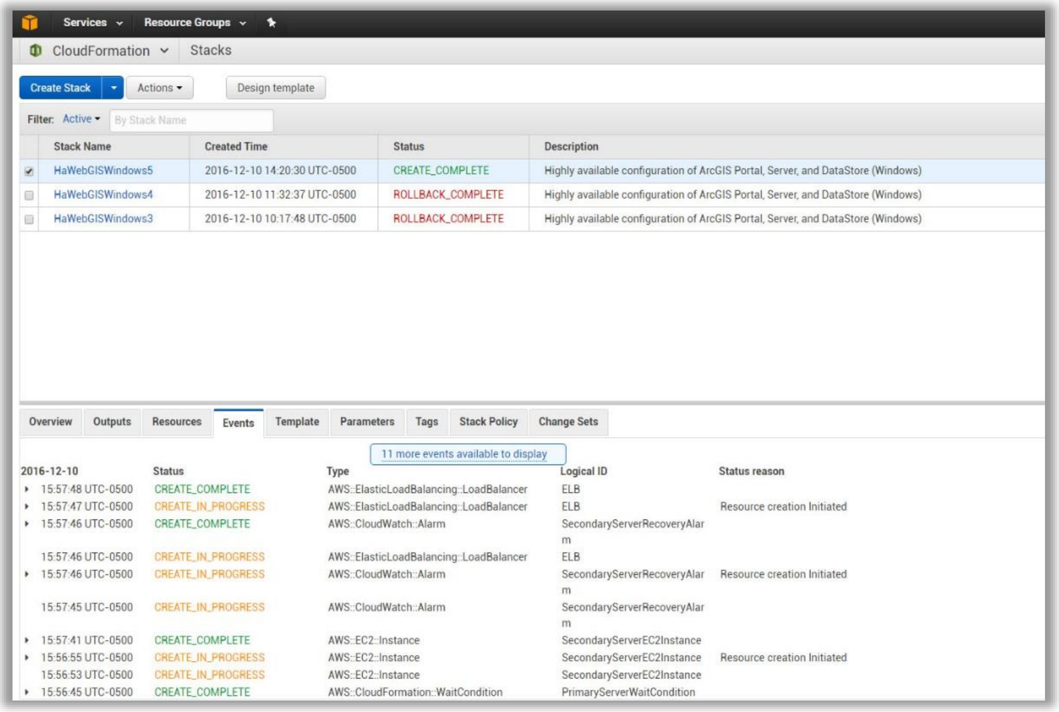


Stack Name	Created Time	Status	Description
HaWebGISWindows3	2016-12-10 10:17:48 UTC-0500	CREATE_IN_PROGRESS	Highly available configuration of ArcGIS Portal, Server, and DataStore (Windows)
HaWebGISWindows	2016-12-05 18:48:53 UTC-0500	ROLLBACK_COMPLETE	Highly available configuration of ArcGIS Portal, Server, and DataStore (Windows)

2016-12-10	Status	Type	Logical ID	Status reason
10:19:17 UTC-0500	CREATE_COMPLETE	AWS::EC2::SubnetRouteTableAssociation	Subnet2RouteTableAssociation	
10:19:12 UTC-0500	CREATE_COMPLETE	AWS::EC2::Route	Route	
10:19:12 UTC-0500	CREATE_COMPLETE	AWS::EC2::SubnetRouteTableAssociation	Subnet1RouteTableAssociation	
10:19:01 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SubnetRouteTableAssociation	Subnet2RouteTableAssociation	Resource creation Initiated
10:19:01 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SubnetRouteTableAssociation	Subnet2RouteTableAssociation	
10:18:57 UTC-0500	CREATE_COMPLETE	AWS::EC2::SecurityGroupIngress	SecurityGroupIngress	
10:18:57 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Route	Route	Resource creation Initiated
10:18:56 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SecurityGroupIngress	SecurityGroupIngress	Resource creation Initiated
10:18:56 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SubnetRouteTableAssociation	Subnet1RouteTableAssociation	Resource creation Initiated
10:18:56 UTC-0500	CREATE_COMPLETE	AWS::EC2::Subnet	Subnet2	
10:18:56 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SecurityGroupIngress	SecurityGroupIngress	
10:18:55 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Route	Route	
10:18:55 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SubnetRouteTableAssociation	Subnet1RouteTableAssociation	

This process could take upwards of an hour.

The AWS console will inform you when the stack is complete.

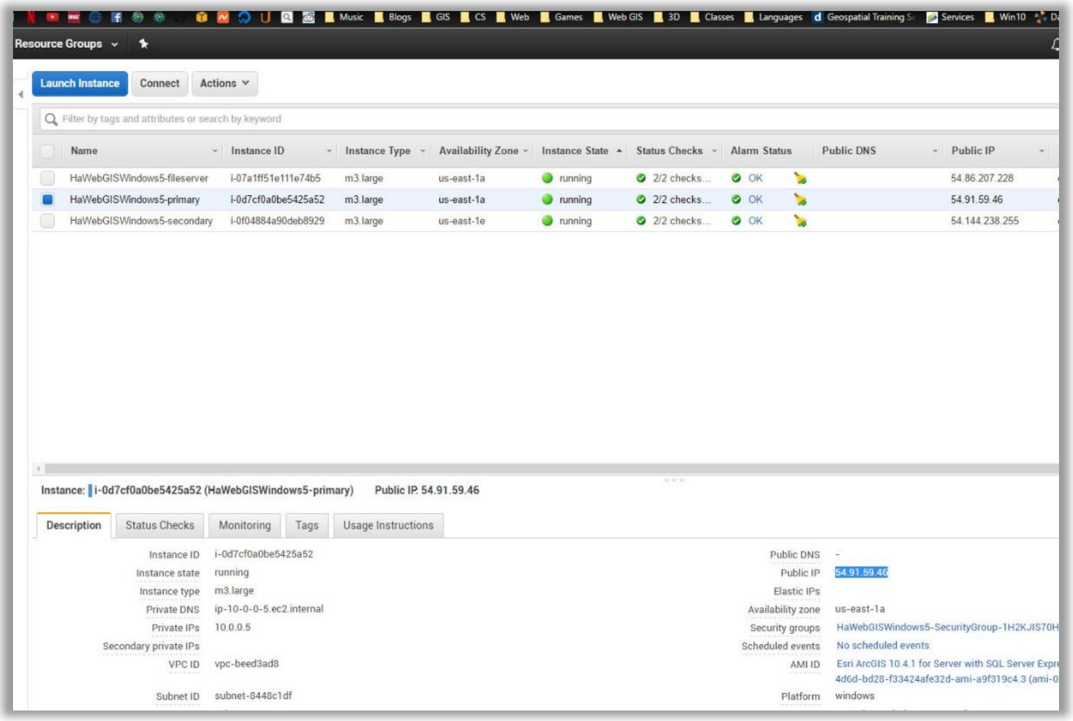


Stack Name	Created Time	Status	Description
HaWebGISWindows5	2016-12-10 14:20:30 UTC-0500	CREATE_COMPLETE	Highly available configuration of ArcGIS Portal, Server, and DataStore (Windows)
HaWebGISWindows4	2016-12-10 11:32:37 UTC-0500	ROLLBACK_COMPLETE	Highly available configuration of ArcGIS Portal, Server, and DataStore (Windows)
HaWebGISWindows3	2016-12-10 10:17:48 UTC-0500	ROLLBACK_COMPLETE	Highly available configuration of ArcGIS Portal, Server, and DataStore (Windows)

11 more events available to display

2016-12-10	Status	Type	Logical ID	Status reason
15:57:48 UTC-0500	CREATE_COMPLETE	AWS::ElasticLoadBalancing::LoadBalancer	ELB	
15:57:47 UTC-0500	CREATE_IN_PROGRESS	AWS::ElasticLoadBalancing::LoadBalancer	ELB	Resource creation Initiated
15:57:46 UTC-0500	CREATE_COMPLETE	AWS::CloudWatch::Alarm	SecondaryServerRecoveryAlarm	
15:57:46 UTC-0500	CREATE_IN_PROGRESS	AWS::ElasticLoadBalancing::LoadBalancer	ELB	
15:57:46 UTC-0500	CREATE_IN_PROGRESS	AWS::CloudWatch::Alarm	SecondaryServerRecoveryAlarm	Resource creation Initiated
15:57:45 UTC-0500	CREATE_IN_PROGRESS	AWS::CloudWatch::Alarm	SecondaryServerRecoveryAlarm	
15:57:41 UTC-0500	CREATE_COMPLETE	AWS::EC2::Instance	SecondaryServerEC2Instance	
15:56:55 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	SecondaryServerEC2Instance	Resource creation Initiated
15:56:53 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	SecondaryServerEC2Instance	
15:56:45 UTC-0500	CREATE_COMPLETE	AWS::CloudFormation::WaitCondition	PrimaryServerWaitCondition	

If you check on your EC2 console you should now see three running instances an auto scaling group and an ELB. All of which have had software pre-installed and configured on them.

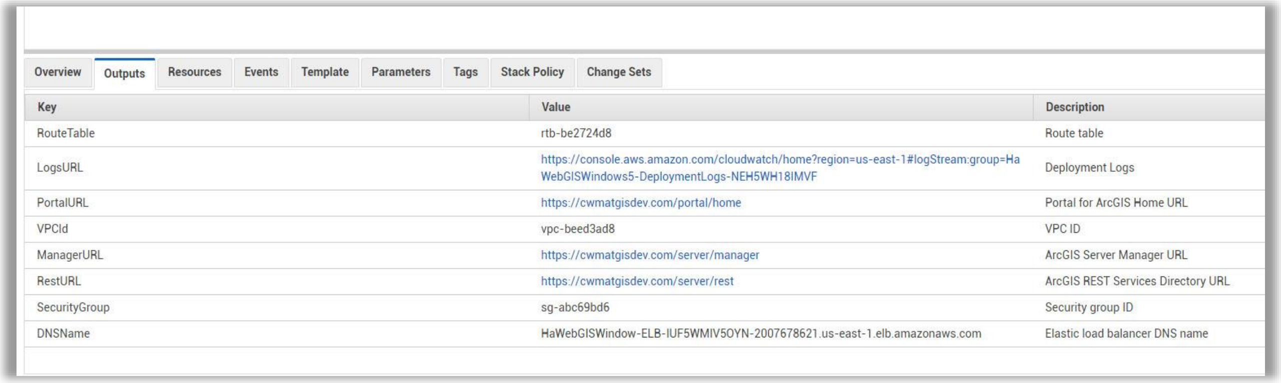


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
HaWebGISWindows5-fileserver	i-07a1ff51e111e74b5	m3.large	us-east-1a	running	2/2 checks...	OK		54.86.207.228
HaWebGISWindows5-primary	i-0d7cf0a0be5425a52	m3.large	us-east-1a	running	2/2 checks...	OK		54.91.59.46
HaWebGISWindows5-secondary	i-0f04884a90deb8929	m3.large	us-east-1e	running	2/2 checks...	OK		54.144.238.255

Instance: i-0d7cf0a0be5425a52 (HaWebGISWindows5-primary) Public IP: 54.91.59.46

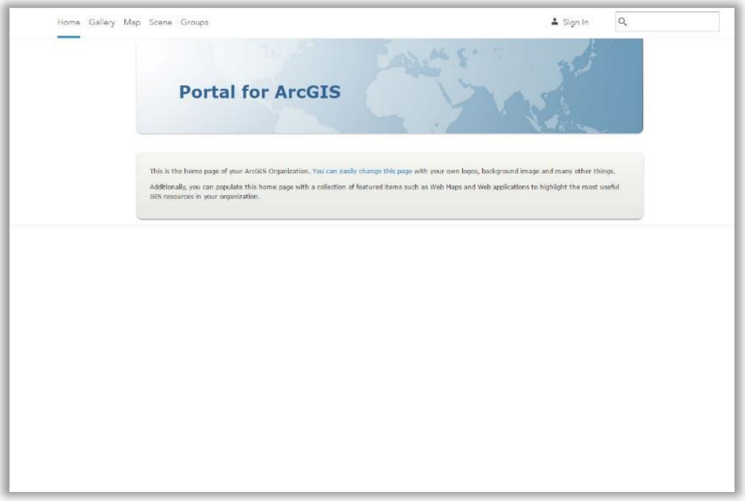
Instance ID	i-0d7cf0a0be5425a52	Public DNS	-
Instance state	running	Public IP	54.91.59.46
Instance type	m3.large	Elastic IPs	
Private DNS	ip-10-0-0-5.ec2.internal	Availability zone	us-east-1a
Private IPs	10.0.0.5	Security groups	HaWebGISWindows5-SecurityGroup-1H2KJIS70H
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-beed3ad8	AMI ID	Esri ArcGIS 10.4.1 for Server with SQL Server Expre 4d6d-bd28-f33424afe32d-ami-a9f319c4.3 (ami-0
Subnet ID	subnet-8448c1df	Platform	windows

Back on the CloudFormation stack details, if we go to the output tab we can see the various links that the stack has set up for us to use our new web GIS resources:



Key	Value	Description
RouteTable	rtb-be2724d8	Route table
LogsURL	https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logStream:group=HaWebGISWindows5-DeploymentLogs-NEH5WH18IMVF	Deployment Logs
PortalURL	https://cwmatgisdev.com/portal/home	Portal for ArcGIS Home URL
VPCId	vpc-beed3ad8	VPC ID
ManagerURL	https://cwmatgisdev.com/server/manager	ArcGIS Server Manager URL
RestURL	https://cwmatgisdev.com/server/rest	ArcGIS REST Services Directory URL
SecurityGroup	sg-abc69bd6	Security group ID
DNSName	HaWebGISWindow-ELB-IUF5WMIV5OYN-2007678621.us-east-1.elb.amazonaws.com	Elastic load balancer DNS name

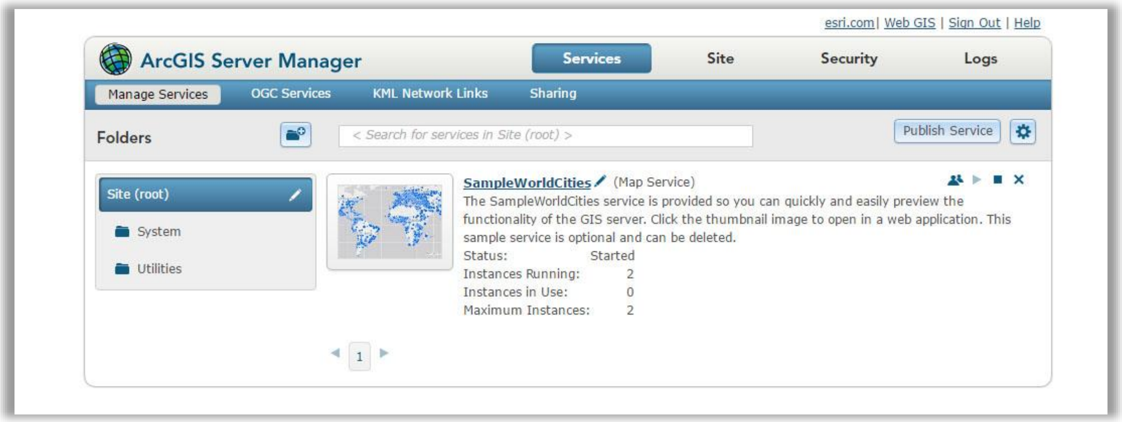
Portal URL: Our ArcGIS Portal site (content management system).



This is the home page of your ArcGIS Organization. You can easily change this page with your own logos, background image and many other things.

Additionally, you can populate this home page with a collection of featured items such as Web Maps and Web applications to highlight the most useful GIS resources in your organization.

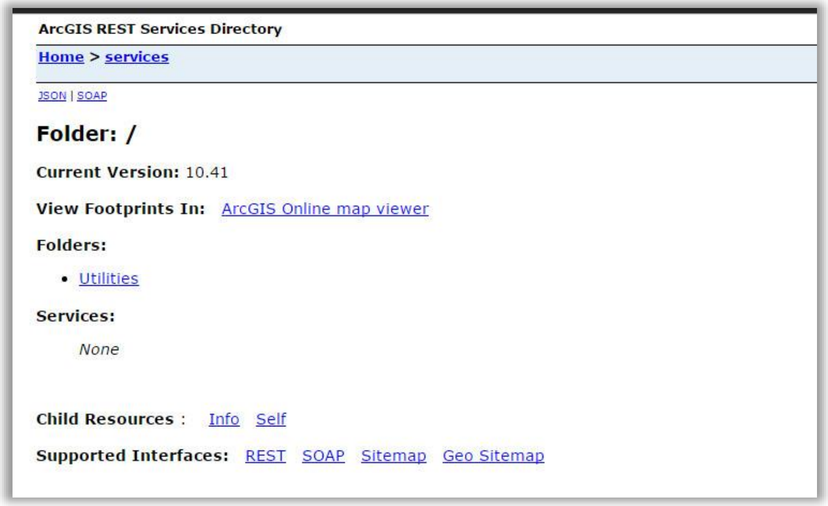
Manager URL: ArcGIS Server manager (work with raw data).



SampleWorldCities (Map Service)
The SampleWorldCities service is provided so you can quickly and easily preview the functionality of the GIS server. Click the thumbnail image to open in a web application. This sample service is optional and can be deleted.

```
Status:               Started
Instances Running:          2
Instances in Use:           0
Maximum Instances:          2
```

Rest URL: Our GIS web services service directory.



ArcGIS REST Services Directory

Home > services

JSON | SOAP

Folder: /

Current Version: 10.41

View Footprints In: ArcGIS Online map viewer

Folders:

- Utilities

Services:

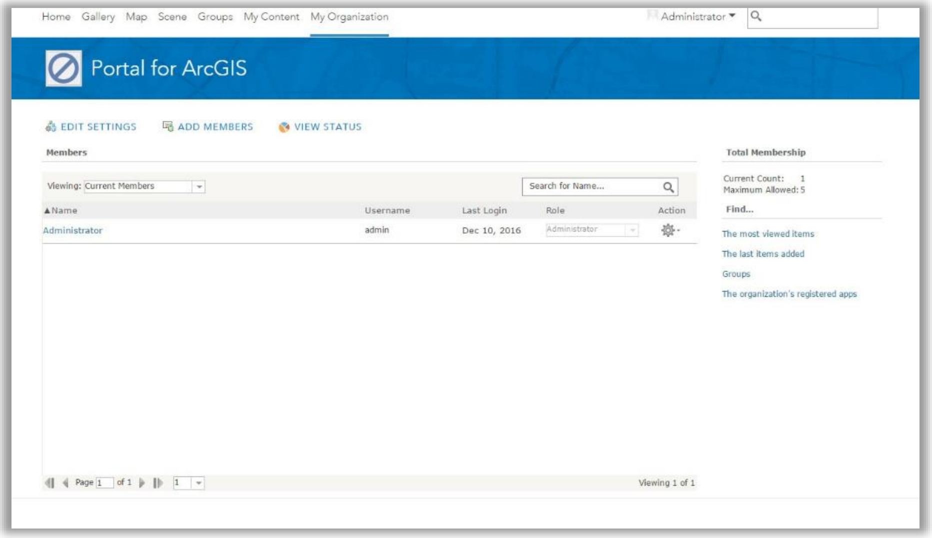
None

Child Resources : Info Self

Supported Interfaces: REST SOAP Sitemap Geo Sitemap

We should now be all set to start adding GIS data.

Just to test, lets sign in to Portal.



Members

Viewing: Current Members

Search for Name...

Name	Username	Last Login	Role	Action
Administrator	admin	Dec 10, 2016	Administrator	

Page 1 of 1

Viewing 1 of 1

Total Membership

Current Count: 1
Maximum Allowed: 5

Find...

The most viewed items
The last items added
Groups
The organization's registered apps

Load data and Publish GIS Web Services

Download Data

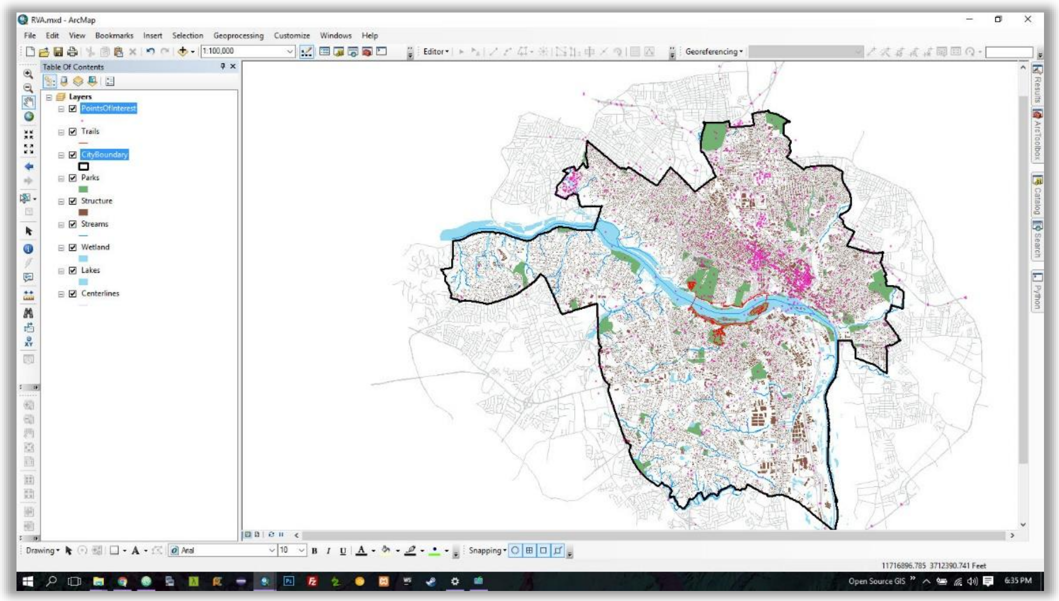
You may use whatever data you like but I will be using data from the City of Richmond, VA's GIS (will be included in the demo folder).

ftp://ftp.ci.richmond.va.us/GIS/Shapefiles/

Stylize

I used ArcMap (ArcGIS Desktop) to symbolize the data layers I downloaded. If you are unfamiliar with ArcMap you can either just use the default colors applied when you pull the data into the table of contents or you can follow the following guide:

http://libraries.mit.edu/files/gis/Symbology.pdf



I used the following layers from Richmond GIS:

- Points of interest
- Trails
- City boundary
- Parks
- Structures (buildings)
- Streams
- Wetlands (which includes the James River – blue part running through the center of the city)
- Lakes
- Road Centerlines

Publish Web Services

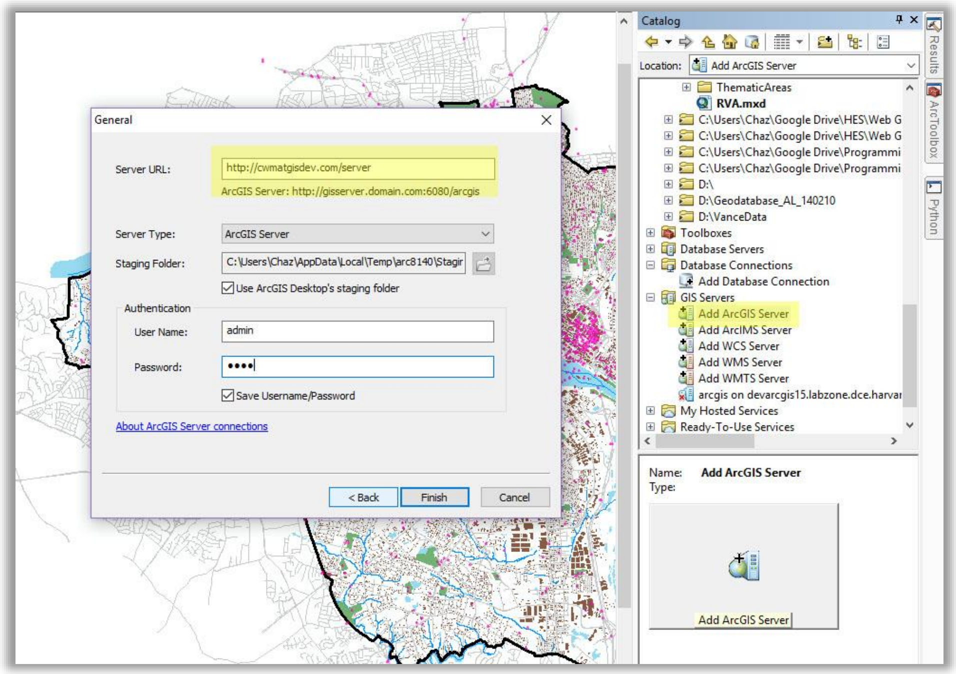
I plan to publish two services:

Web Map Service – A tile cache of images at various map scales portraying the data in our map that is rendered by the server (this is like how you can sometimes see a Google Map loaded into view one square at a time). This method allows for quicker loading than the method mentioned below. We will sue this for our basemap layers.

Web Feature Service – A JSON collection of the actual geometry and attribute data within a dataset which is rendered by the client. This method is slower but gives us access to the attribute data in a dataset (which will enable pop-ups for us). We will use this for our point of interest layers.

Publish a Web Mapping Service

Make a server connection to our AWS hosted ArcGIS Server using the URL from CloudFront (minus the /manager/ - this extension is for accessing our server via a web client rather than a software client like ArcMap which just needs the /server/ part):



General

Server URL: http://cwmatgisdev.com/server

ArcGIS Server: http://gisserver.domain.com:6080/arcgis

Server Type: ArcGIS Server

Staging Folder: C:\Users\Chaz\AppData\Local\Temp\arc8140\Stagir

☑ Use ArcGIS Desktop's staging folder

Authentication

User Name: admin

Password: ••••

☑ Save Username/Password

Now we could publish a service directly using this link. However, for this guide we will be using the ArcPy Python API to programmatically publish GIS data and web services to our AWS instances.

Right click a folder in Arc Catalog and create a new Server Connection. Then follow the steps above to make the connection file.

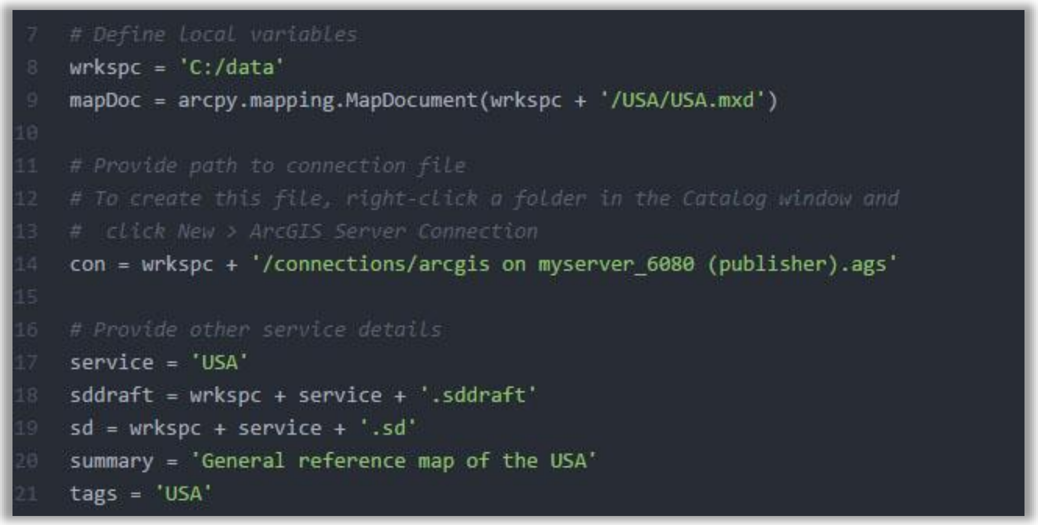
Attached with this guide will be a publish_map_services.py file.

Lines 8-21: Sets up our local variables for the script inputs.

wrkspc = The workspace, aka folder path to where your main working files/folders are located.

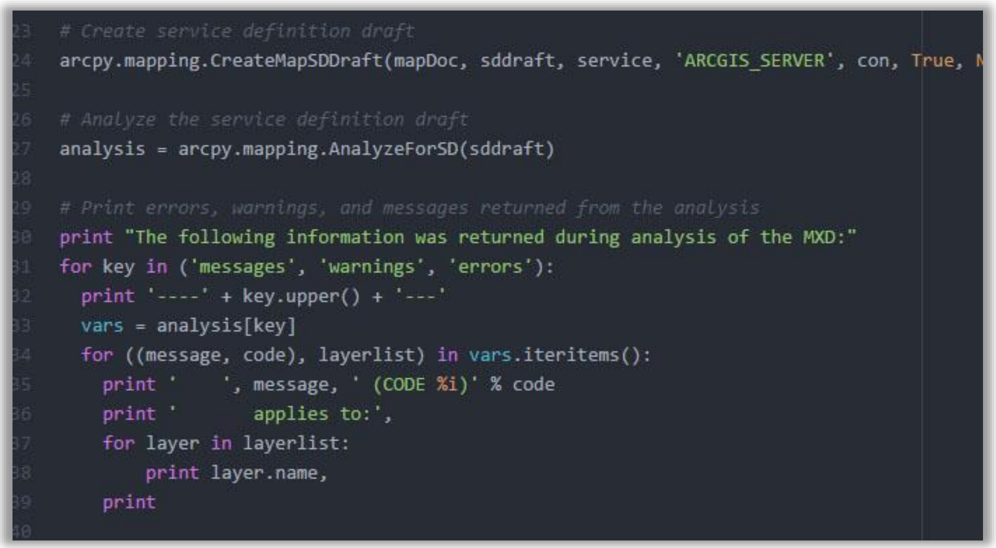
mapDoc = The .mxd file you want to turn into a map service.

con = the connection file from above.



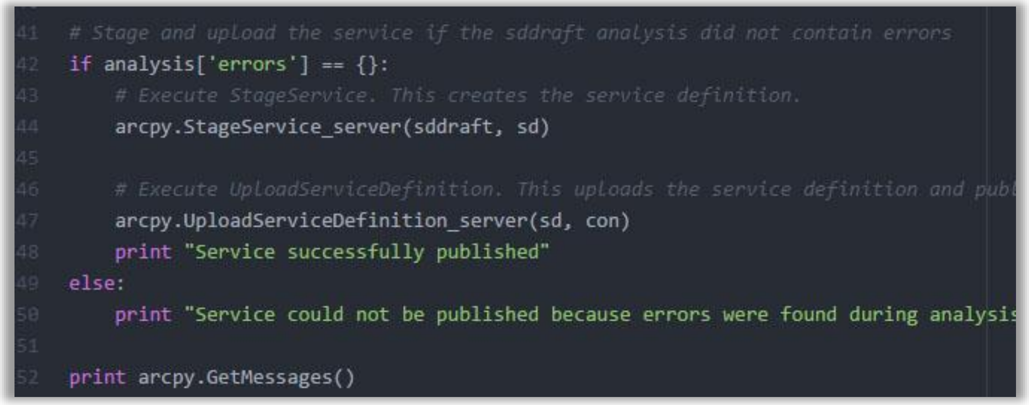
```
7   # Define local variables
8   wrkspc = 'C:/data'
9   mapDoc = arcpy.mapping.MapDocument(wrkspc + '/USA/USA.mxd')
10
11  # Provide path to connection file
12  # To create this file, right-click a folder in the Catalog window and
13  #  click New > ArcGIS Server Connection
14  con = wrkspc + '/connections/arcgis on myserver_6080 (publisher).ags'
15
16  # Provide other service details
17  service = 'USA'
18  sddraft = wrkspc + service + '.sddraft'
19  sd = wrkspc + service + '.sd'
20  summary = 'General reference map of the USA'
21  tags = 'USA'
```

Line 24 initiates the service definition. Lines 27-39 analyses the map document to find any errors before attempting to publish.



```
23  # Create service definition draft
24  arcpy.mapping.CreateMapSDDraft(mapDoc, sddraft, service, 'ARCGIS_SERVER', con, True,
25
26  # Analyze the service definition draft
27  analysis = arcpy.mapping.AnalyzeForSD(sddraft)
28
29  # Print errors, warnings, and messages returned from the analysis
30  print "The following information was returned during analysis of the MXD:"
31  for key in ('messages', 'warnings', 'errors'):
32    print '----' + key.upper() + '---'
33    vars = analysis[key]
34    for ((message, code), layerlist) in vars.iteritems():
35      print '    ', message, ' (CODE %i)' % code
36      print '       applies to:',
37      for layer in layerlist:
38          print layer.name,
39      print
40
```

Lines 42-52 attempt to publish the service.



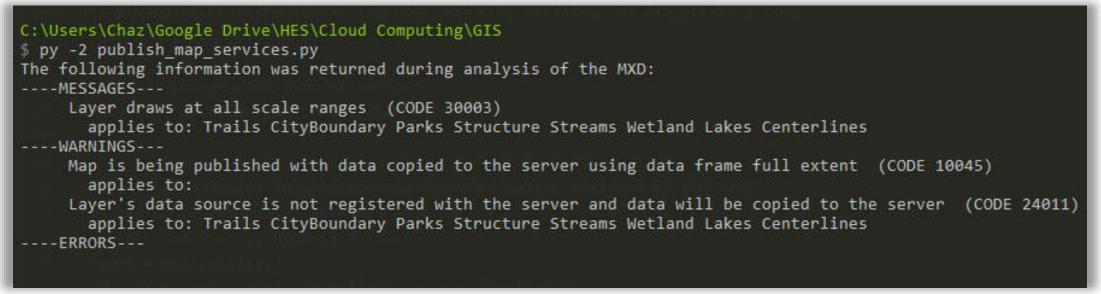
```
# Stage and upload the service if the sddraft analysis did not contain errors
if analysis['errors'] == {}:
    # Execute StageService. This creates the service definition.
    arcpy.StageService_server(sddraft, sd)

    # Execute UploadServiceDefinition. This uploads the service definition and publ
    arcpy.UploadServiceDefinition_server(sd, con)
    print "Service successfully published"
else:
    print "Service could not be published because errors were found during analysis

print arcpy.GetMessages()
```

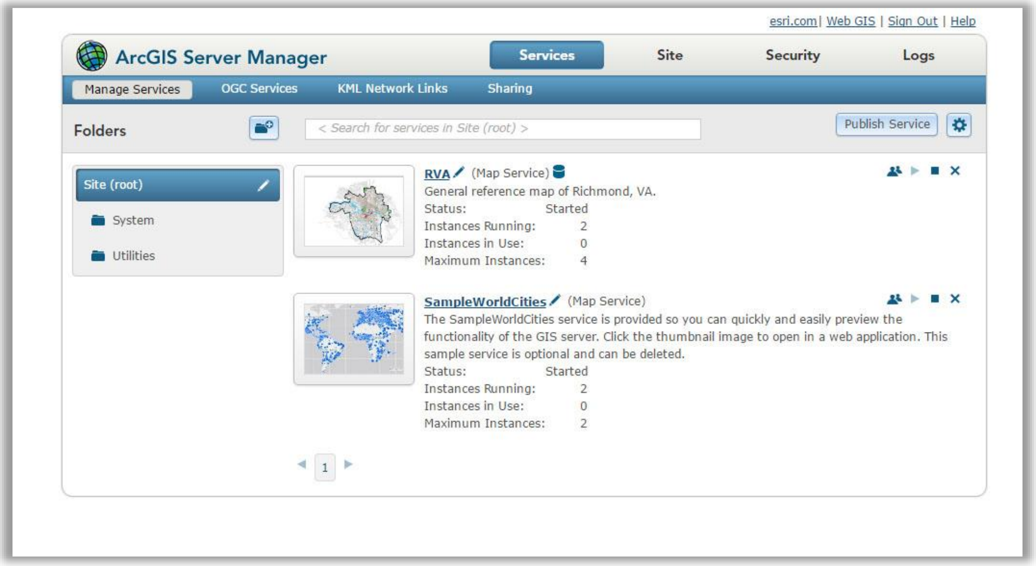
Update the parameters and run the script.

A series of messages and maybe some warnings will come up. If no errors show up the script should eventually finish with a success message.



```
C:\Users\Chaz\Google Drive\HES\Cloud Computing\GIS
$ py -2 publish_map_services.py
The following information was returned during analysis of the MXD:
----MESSAGES---
     Layer draws at all scale ranges  (CODE 30003)
       applies to: Trails CityBoundary Parks Structure Streams Wetland Lakes Centerlines
----WARNINGS---
     Map is being published with data copied to the server using data frame full extent  (CODE 10045)
       applies to:
     Layer's data source is not registered with the server and data will be copied to the server  (CODE 24011)
       applies to: Trails CityBoundary Parks Structure Streams Wetland Lakes Centerlines
----ERRORS---
```

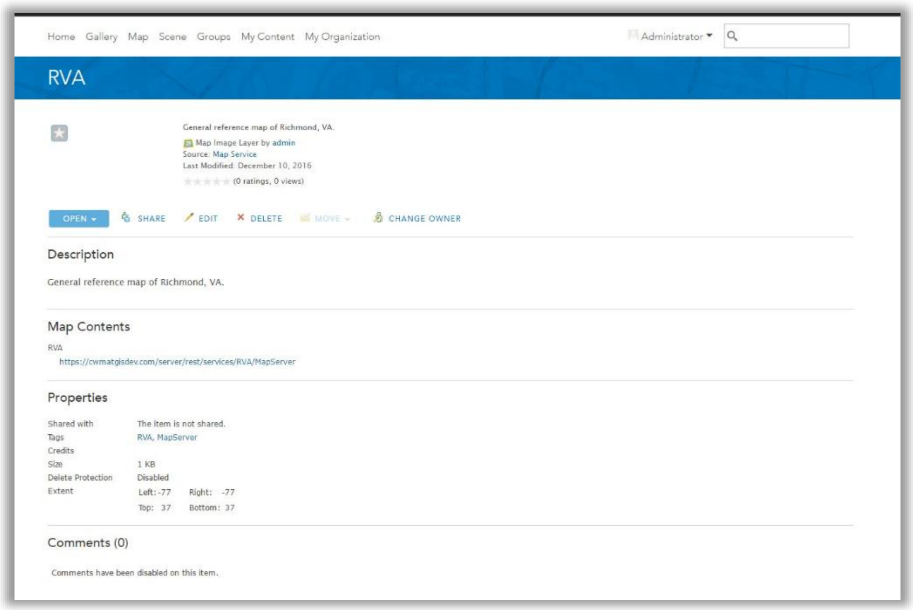
We can check Server Manager and see that our RVA service has been published:



RVA (Map Service)
General reference map of Richmond, VA.
Status: Started
Instances Running: 2
Instances in Use: 0
Maximum Instances: 4

SampleWorldCities (Map Service)
The SampleWorldCities service is provided so you can quickly and easily preview the functionality of the GIS server. Click the thumbnail image to open in a web application. This sample service is optional and can be deleted.
Status: Started
Instances Running: 2
Instances in Use: 0
Maximum Instances: 2

Also, we can check Portal and see the same.



RVA

General reference map of Richmond, VA.
Map Image Layer by admin
Source: Map Service
Last Modified: December 10, 2016
(0 ratings, 0 views)

Description

General reference map of Richmond, VA.

Map Contents

RVA
https://cwmatgisdev.com/server/rest/services/RVA/MapServer

Properties

Shared with	The item is not shared.
Tags	RVA, MapServer
Credits	
Size	1 KB
Delete Protection	Disabled
Extent	Left: -77 Right: -77 Top: 37 Bottom: 37

Comments (0)

Comments have been disabled on this item.

Note that CloudFormation automatically defaults storage of uploaded data and service definitions to the EBS D: drive for our GIS Server instances.

Publish a Web Feature Service

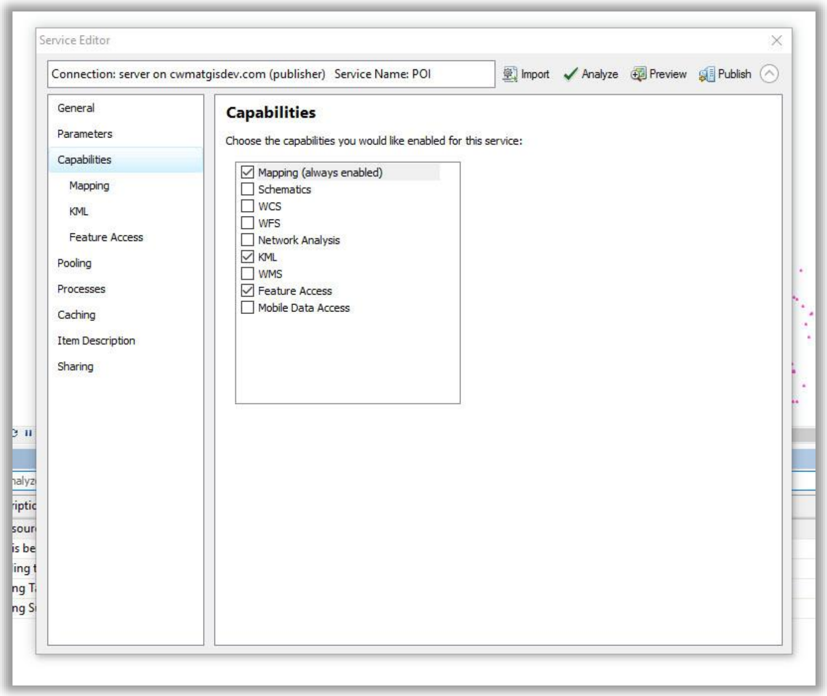
We can use the ArcPy API again here but to publish a feature service we must make a service definition by hand and then pass that to the API. For our purposes this is a bit unnecessary as once the service definition is created we are just one button away from publishing the service.

So, here is a quick overview of how to publish this service manually.

While in the map document you want to publish as a service:

Select File > Share as a Service > Publish a Service > Select our CloudFormation GIS Server > Save in root.

A wizard will pop up. You can keep all the defaults except check the Feature Access check box in the capabilities tab. Then click publish.


Service Editor

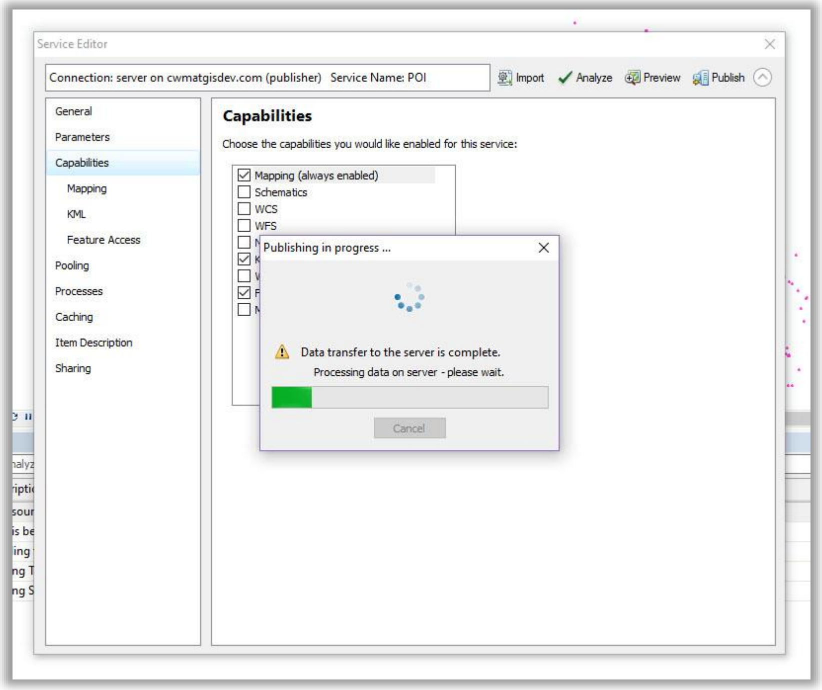
Connection: server on cwmatgisdev.com (publisher) Service Name: POI

Capabilities

Choose the capabilities you would like enabled for this service:

- [x] Mapping (always enabled)
- [ ] Schematics
- [ ] WCS
- [ ] WFS
- [ ] Network Analysis
- [x] KML
- [ ] WMS
- [x] Feature Access
- [ ] Mobile Data Access

Again, the data will transfer to the AWS instances:

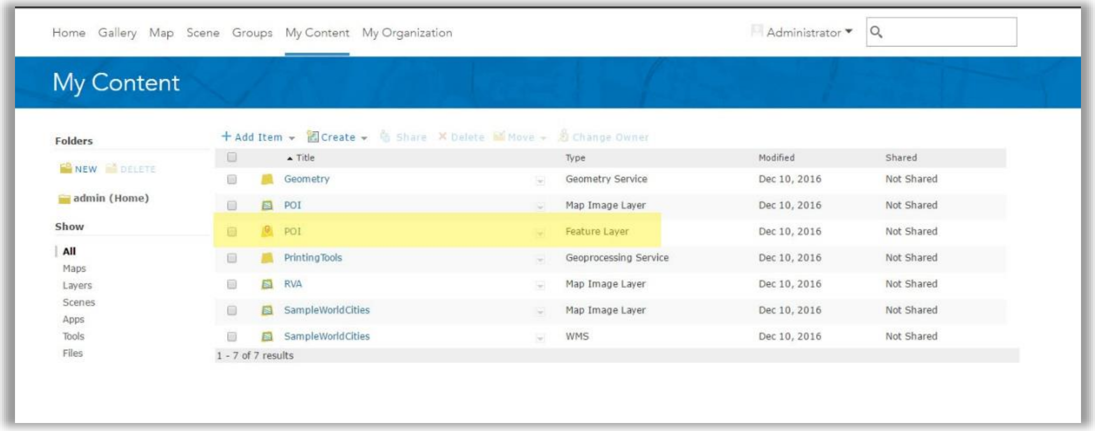

Service Editor

Connection: server on cwmatgisdev.com (publisher) Service Name: POI

Publishing in progress ...

Data transfer to the server is complete.
Processing data on server - please wait.

Check your portal and see that it has been successfully uploaded.



Home Gallery Map Scene Groups My Content My Organization Administrator

My Content

Folders: NEW DELETE; admin (Home)

Show: All, Maps, Layers, Scenes, Apps, Tools, Files

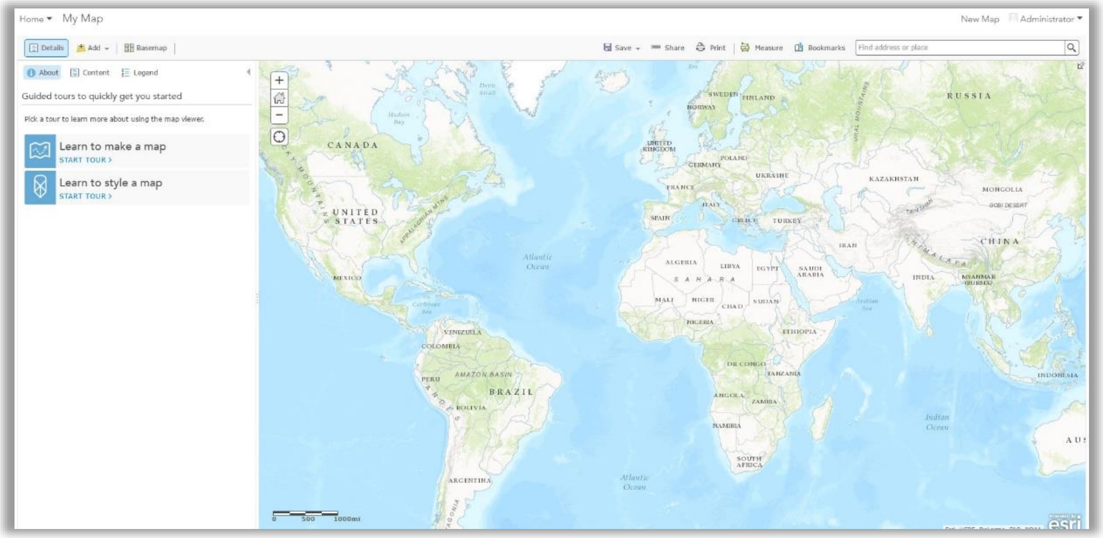
+ Add Item, Create, Share, Delete, Move, Change Owner

Title	Type	Modified	Shared
Geometry	Geometry Service	Dec 10, 2016	Not Shared
POI	Map Image Layer	Dec 10, 2016	Not Shared
POI	Feature Layer	Dec 10, 2016	Not Shared
PrintingTools	Geoprocessing Service	Dec 10, 2016	Not Shared
RVA	Map Image Layer	Dec 10, 2016	Not Shared
SampleWorldCities	Map Image Layer	Dec 10, 2016	Not Shared
SampleWorldCities	WMS	Dec 10, 2016	Not Shared

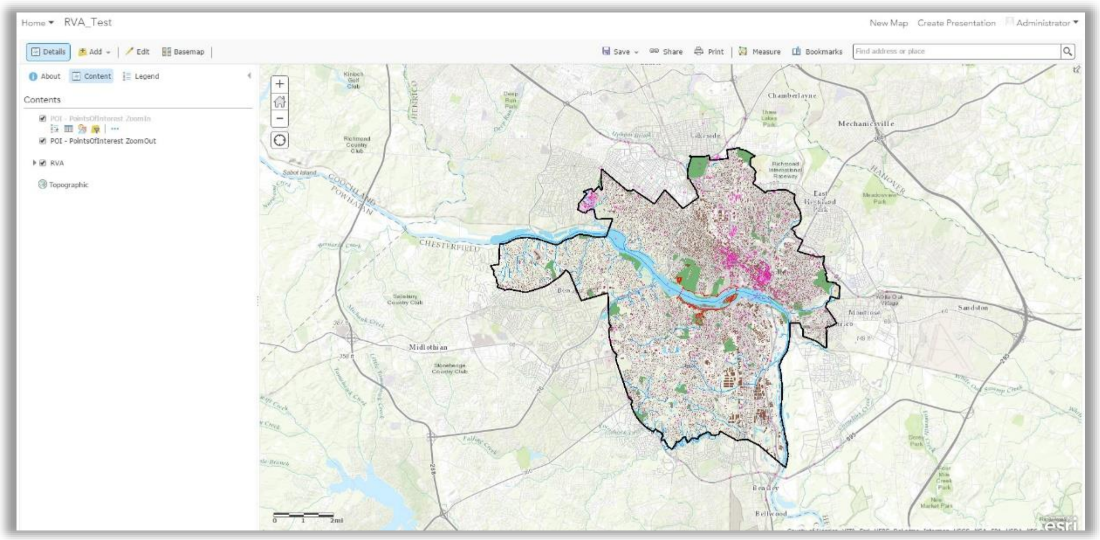
1 - 7 of 7 results

Make a Web Map

Go to Portal > Map.



Add some layers by: Add > Search for Layers > In: Portal > Add the layers we uploaded.



Conclusion

Now we have a simple map using GIS web services hosted on our AWS cloud. We tweaked JSON templates to customize the ArcGIS stack we wanted to use and utilized Python and the AWS Python SDK, Boto, to stand up our infrastructure. We then used Python and the ArcPy API to upload data and publish a webs service to our cloud instances. Lastly, for fun, we created a web map using the map builder software that comes with our ArcGIS Portal instance. If we wanted to dig in further, we could use the ArcGIS JavaScript API to make a more customized web map. This is beyond the scope of this project though, but to learn more go to: https://developers.arcgis.com/javascript/

I hope you have enjoyed this tutorial.

References

https://aws.amazon.com/cloudformation/

https://www.python.org/

https://pbs.twimg.com/profile_images/479487015621111809/53mx3P5l.jpeg

https://aws.amazon.com/sdk-for-python/

https://developers.arcgis.com/javascript/

http://stackoverflow.com/questions/6307886/how-to-create-pfx-file-from-certificate-and-private-key

http://www.richmondgov.com/GIS/

http://libraries.mit.edu/files/gis/Symbology.pdf

http://server.arcgis.com/en/server/10.4/administer/windows/example-publish-a-map-service-from-a-map-document-mxd-.htm

https://langui.sh/2009/01/24/generating-a-pkcs12-pfx-via-openssl/