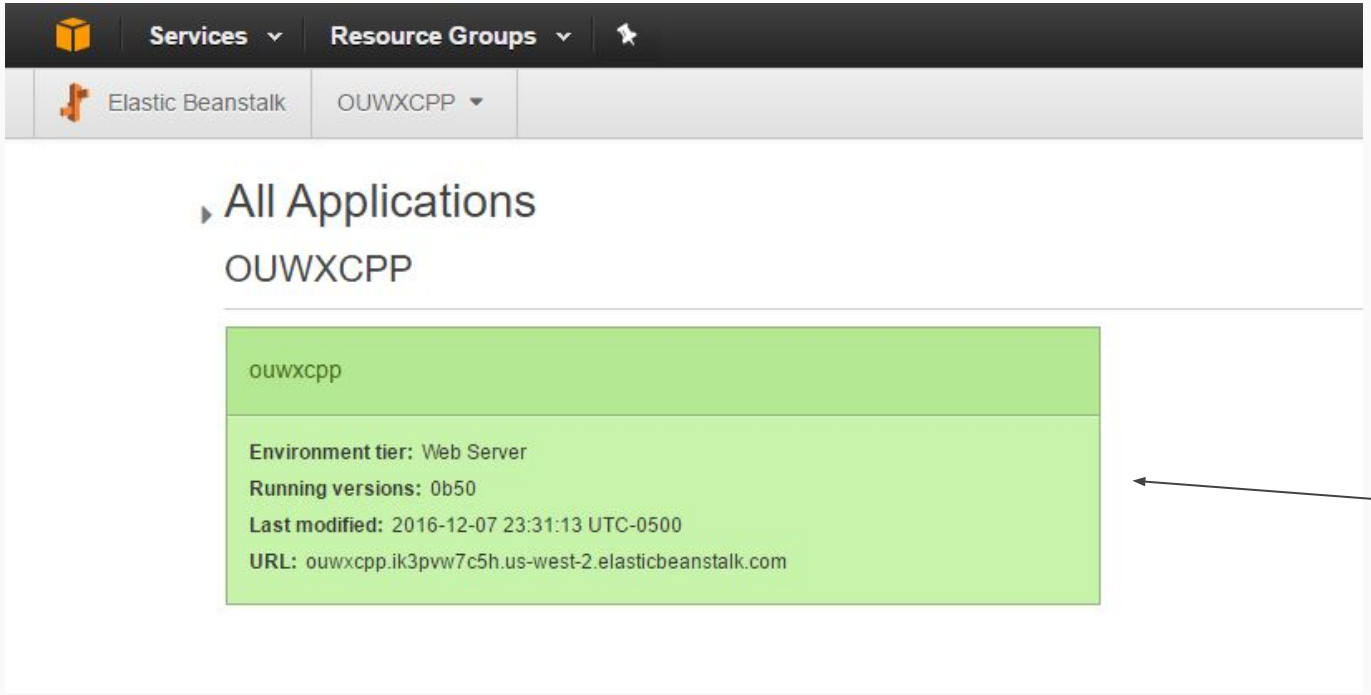
AWS Elastic Beanstalk Command Line Interface

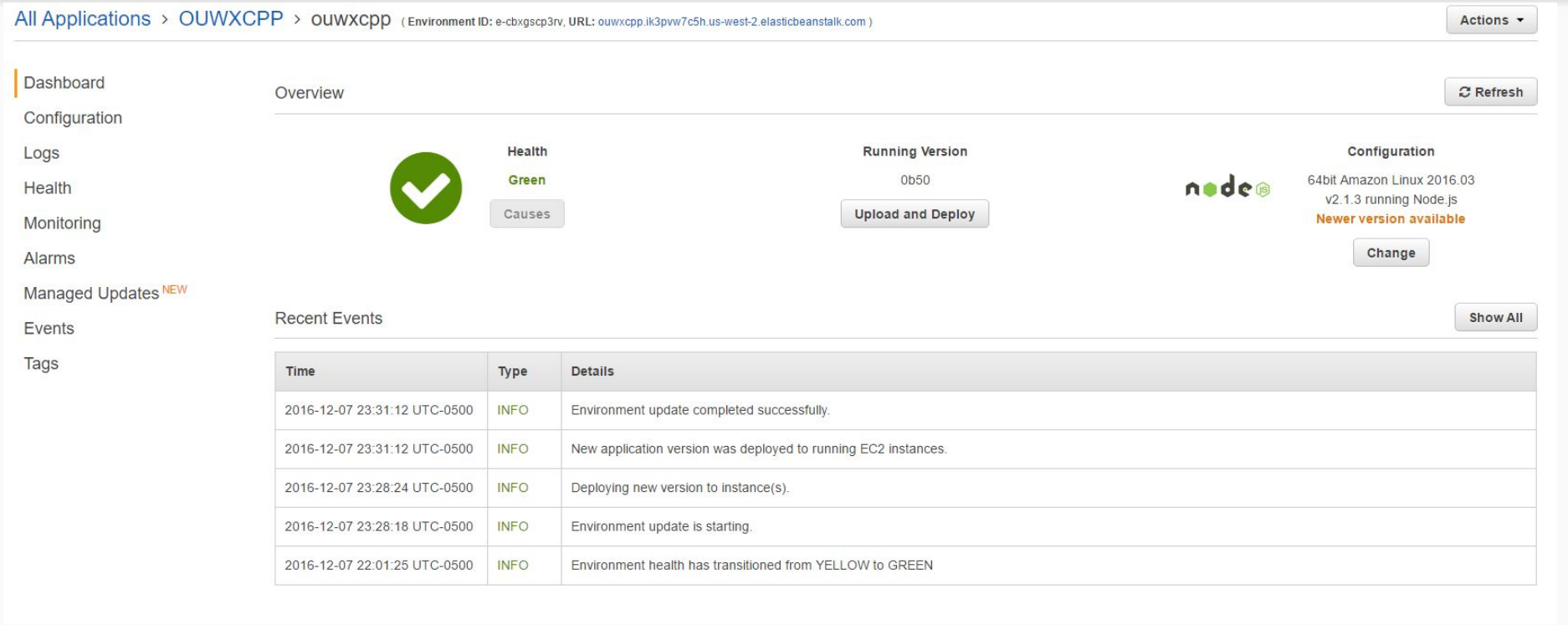
Robert Whitmer

Background on EB - Application/Environment Interface



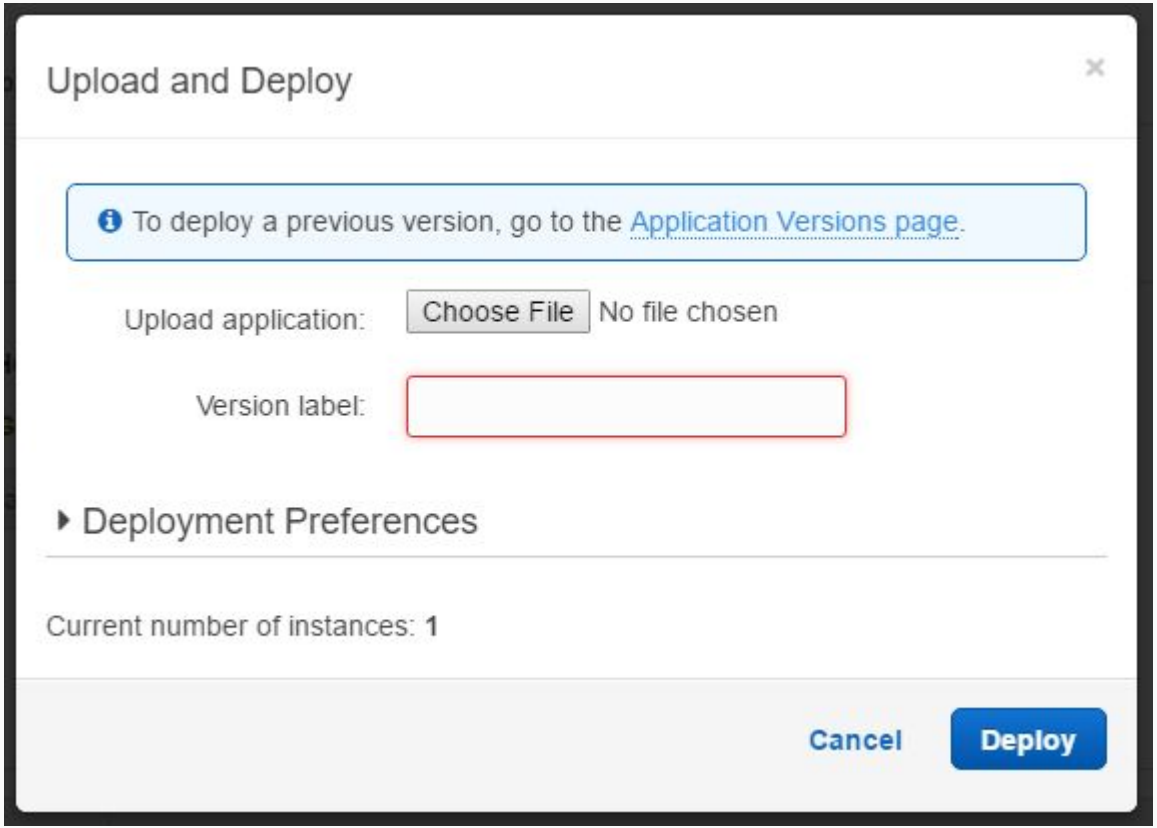
Each application can have multiple environments (dev, test, prod, etc)

EB Web-interface



While the web interface has many useful tools as shown above, it lacks in ease of deployment.

Upload and Deploy



The web interface for the deployment requires your project be in a zipped archive with no top level folder.

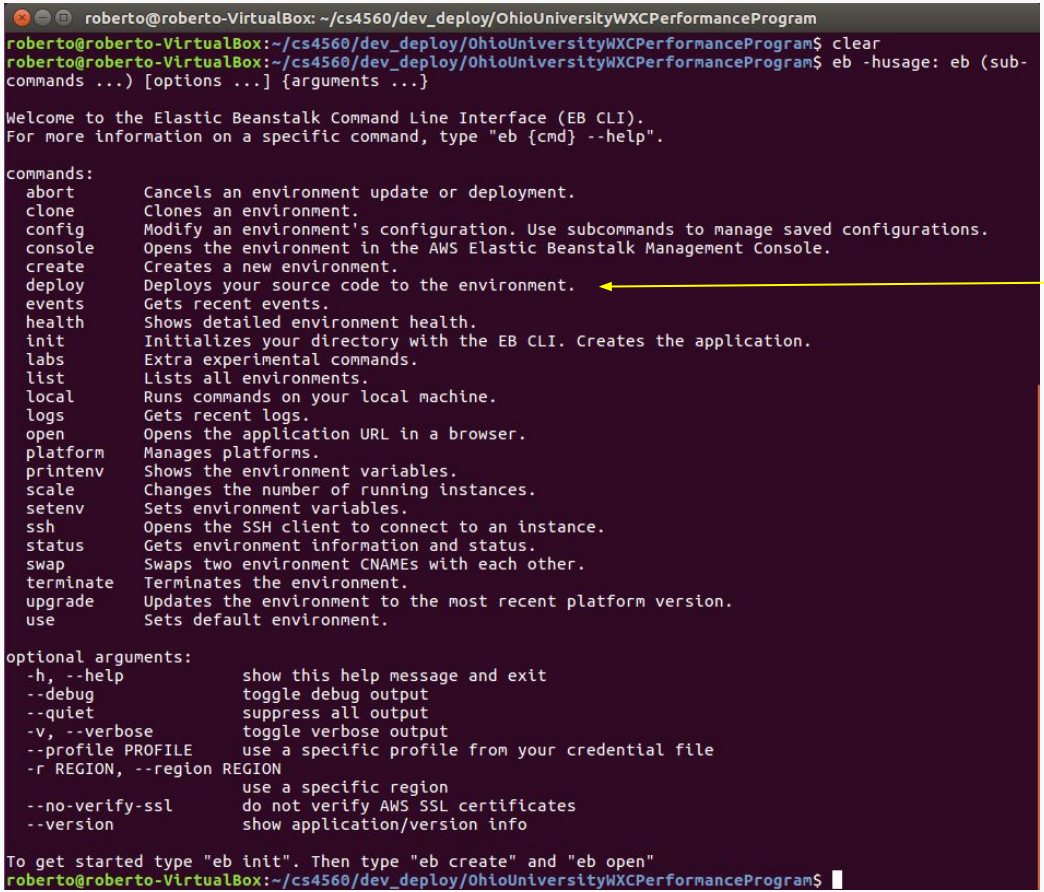
Doing this every time you want to deploy can be tedious and cause errors depending on the structure of your project.

AWS Elastic Beanstalk CLI

On the other side, the command line interface allows for

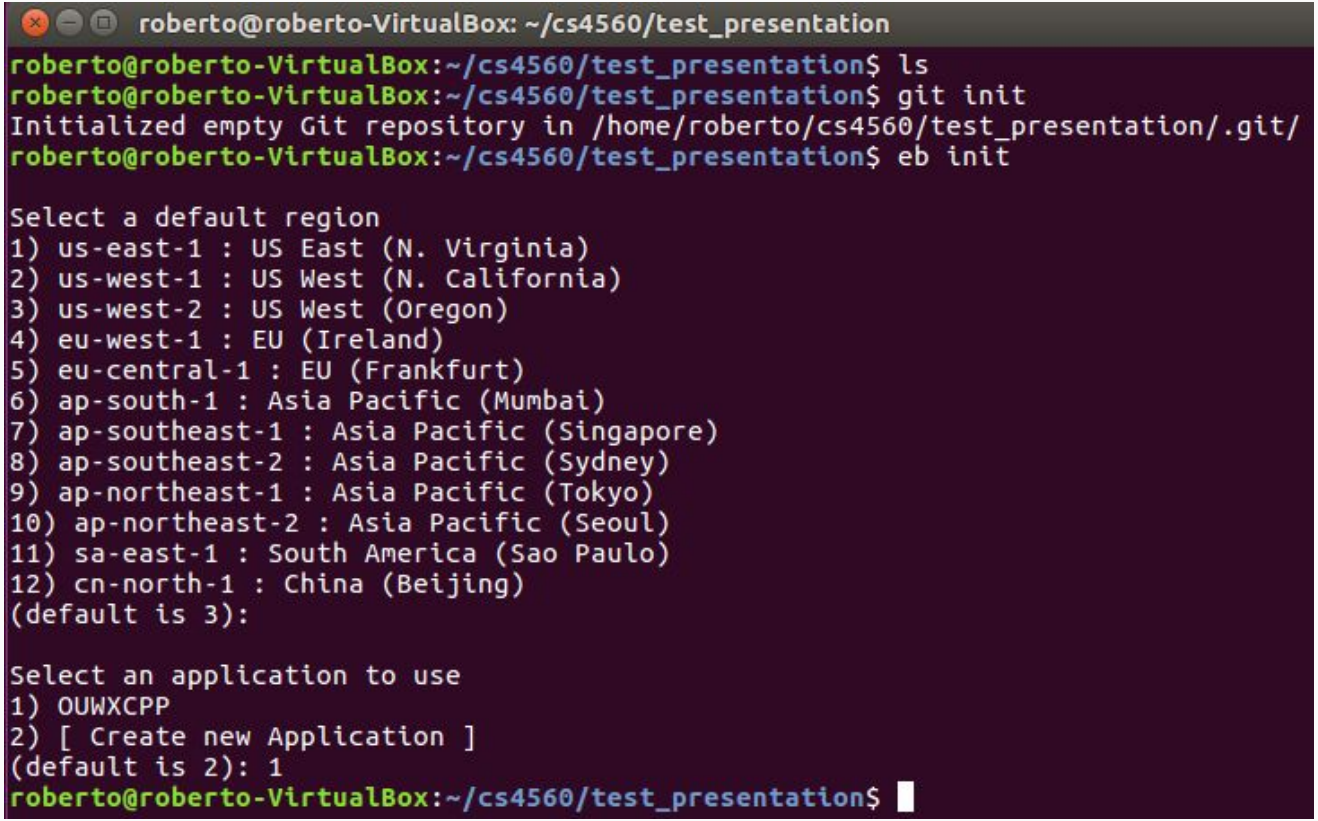
- Rapid deployment without need to worry about archiving project folder
- Scripted utilities for managing the environment based on health or other factors
- **Connection with a git repository**

Command line options



The “eb deploy” command instantly deploys your project in the current directory to the live environment, no archiving necessary.

Connection with Git



As long as there was an initialized git repo in the directory before “eb init” was ran, the ebcli will be able to connect with git.

eb deploy & eb health

```
roberto@roberto-VirtualBox:~/cs4560/dev_deploy/OhioUniversityWXCPerformanceProgram$ eb deploy
Creating application version archive "app-c6ca-170117_174238".
Uploading OUWXCPP/app-c6ca-170117_174238.zip to S3. This may take a while.
Upload Complete.
INFO: Environment update is starting.
INFO: Deploying new version to instance(s).
INFO: New application version was deployed to running EC2 instances.
INFO: Environment update completed successfully.
```

```
ouwxcpp                              Unknown                    2017-01-20 00:04:05
instances: 1 Total, 1 InService, 0 Other
 Status: Ready Health Green

  instance-id        EC2 Health      ELB State       ELB description          health
  i-0213dda4559d69cc3 running        InService       N/A
```

Clearly, all of the web tools can be matched here, and with more possibilities!

Connecting with Git

Running “eb deploy” will deploy the version of the project that is in your local repository. That is, what has been committed **locally**.

This allows a convenient development feature to be exposed:

You can develop and test locally and deploy changes (potentially to a test/dev environment, not in our case), while still maintaining the integrity of the remote repository so you don’t affect others’ development. If something breaks on the deployment, you can simply revert the changes. If it works on the live version, you can then push out your changes to the remote repository.

Scripting Possibilities

Because these features are on the command line, they can easily be set up in scripts to automatically do these things. For a simple example, I created a short script that commits my changes locally and deploys to the live environment. If I find that the live version has issues for some reason, another script could be used to revert the environment to the most recent version and then reset my local repository to the previous state.

Script Example

```
#!/usr/bin/env bash


if [ "$#" -ne 1 ]; then
    echo "Error: Usage: ./deploy.sh \"COMMIT MESSAGE\""
fi

commit=$1
gcommand="git commit -m "\""$commit"\"
echo $gcommand

git add .
eval $gcommand

eb deploy
```

Help Document for Setup

I made the following for our team to help them set up the EB CLI.

Tutorial for setting up EB CLI w/ Git.
- Robert Whitmer
---
1. This assumes you have some UNIX terminal equivalent (cygwin, virtualbox, mac, nitrous, etc).
   - With that, you'll probably want a package manager like apt-get.
2. Install pip (python installer) to allow easy setup of EB CLI.
   - Depending on your system, sudo apt-get install python-pip should work
3. run 'pip install awsebcli'
4. Clone out the git repo from our Project Github.
5. run 'eb init'
   - You will need to input your Access Key and Secret Access Key (in the .txt file on slack I sent a while back)
   - Select the region (Oregon is where the main application is, if you pick n. viriginia you won't be able to access the live deployment)
   - Select the application (OhioUniversityWXCPerformanceProgram is the 'live' one)
   - If you run 'eb list' you should see the environment (only one for now -- dev).
6. You should be able to then make changes, and commit them to the git repo (not necessary to push to be able to deploy, but obviously don't forget to push out if the changes didn't break anything)
7. Once commited, run 'eb deploy' to deploy the application. If it broke the deployment, you can always git reset HEAD to revert back (if you didn't push out already).
8. Bada boom bada bang you should be good to go!

My general plan for development is to do coding/testing locally and then commit to git once tested. Then deploy to the server and see if it actually runs live, if so, great, push the changes to the rest of the team. If it broke things, revert the commit and test more locally. This assumes you're committing from the command line, as it's not so easy to revert on the web interface.