AWS Production Servers

ALL HOW TOs

References:

- Article: Best Practices for Rails app using Elasticbeanstalk
- Article: Update Bundler on AWS Elastic Beanstalk

AWS Console All Services Compute Elastic Beanstalk change Ohio to N. Virginia in top menu

AWS Console Storage S3 Buckets search for app name

Logs

click Logs in left side menu

- fetch full logs
- download and unzip
- interesting logs

log/eb-commandprocessor.log	shows what happened during deploy
log/nginx/access.log	shows URLs accessed for qa_server
app/containerfiles/logs/production.log	rails app log

```
eb logs      # equivalent to tail -100
eb logs -a   # download all logs to local directory  in rails app .elasticbeanstalk/l
```

ssh to AWS

- eb init
- eb use _machine_name_
- eb ssh

- locations of interest

path	purpose	ex files
/var/app/current	navigate to app	
/opt/elasticbeanstalk/hooks/appdeploy/pre	scripts to run before deploy starts	10_bundle_install.sh
/opt/elasticbeanstalk/hooks/appdeploy/post	scripts to run after after deploy starts	01_rails_support.sh
/opt/elasticbeanstalk/support	EB_SUPPORT_DIR	

/opt/elasticbeanstalk/support/scripts	EB_SCRIPT_DIR	
/var/app/ondeck	EB_APP_STAGING_DIR	
webapp	EB_APP_USER	not a location, but used in 10_bundle_install.sh

- connecting to the database
    - ```
      $ sudo mysql -h _DATABASE_HOST_  -u _DATABASE_RAILS_USER_ -p
      Enter password: _DATABASE_RAILS_USER_PW_
      ```

      Values for _DATABASE_* are in the AWS config

Debugging on AWS

- tail logs
    - `tail -f -n 50 log/integration.log`
    - `tail -f -n 50 log/staging.log`
    - `tail -f -n 50 log/production.log`
    - `tail -f -n 50 log/monitor.log`
    - `tail -f -n 50 log/performance_cache.log`
- add puts statements and restart
    - `sudo service httpd restart`
- grep for errors in logs
    - `grep 500 log/production.log`
    - `grep Timeout log/production.log`
    - `grep "RDF::Graph#load failure" log/production.log`

Working with Solr

- get solr location from AWS Console All Services Compute Elastic Beanstalk _MACHINE_NAME_ Configuration Software Modify SOLR_URL remove everything /solr/ on (e.g. remove /solr/exhibits from the end of the URL) paste URL in browser
- check if solr is running

```
ssh _URL_MINUS_HTTP_MINUS_PORT_TO_END_     ### NO PERMISSION TO SSH TO THIS MACHINE
ps ef | grep solr
```

- start if not running or restart solr if running but not responding

```
sudo service solr start
sudo service solr restart
```

Using eb cli

Ref: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb3-ssh.html

Cool commands

command	example	comments
eb init	eb init	• us-east-1 • choose _MACHINE_NAME_-int • choose default environment (will vary) • type **n** when asked about CodeCommit
eb list	eb list	shows all servers you can connect to
eb ssh	eb ssh _MACHINE_NAME_-stg	connect to default or a specific server (see eb list for server names) app at: /var/app/current
eb printenv		
eb setenv _NAME_=_VALUE_		set environment vars from command line
```

aws logs		
aws `elasticbeanstalk restart-app-server`	`aws elasticbeanstalk restart-app-server --environment-name my-env`	

Updating bundler on AWS

Article: Update Bundler on AWS Elastic Beanstalk

In app on laptop (e.g. CUL-IT/qa_server)

- Edit `/.ebextensions/upgrade_bundler.config`
  - make sure ruby version is correct
  - update bundler version
- update app version in `/config/locales/qa_server.en.yml`
- add version changes to `/CHANGELOG.md` (e.g. "update bundler to 2.1.4 on AWS")

At app root in terminal on laptop

- update bundler to desired version (e.g. for 2.1.4 use command: `gem install bundler -v 2.1.4`)
- run any bundle command to update Gemfile.lock's version (e.g. update qa_server gem: `bundle update qa_server`)
  - Gemfile.lock should now say it was built with the new version of bundler. (e.g. last few lines read: `BUNDLED WITH 2.1.4`)

Commit

- .ebextensions/upgrade_bundler.config
- Gemfile.lock
- qa_server.en.yml
- CHANGELOG.md