Securing your DevOps Toolchain in the Cloud

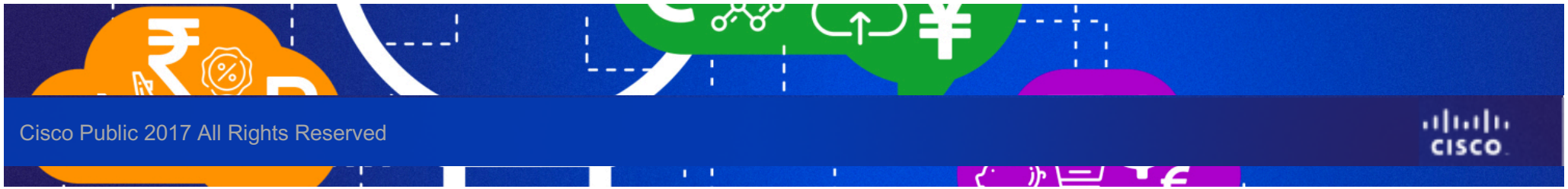
Tony Rice

Cisco Security & Trust Organization

Prerequisites

If you don't already have these established, please take care of this now

1. Amazon AWS account (free tier is fine)
   - aws.amazon.com/free
2. Github.com account



Python Prep (for local testing)

- Install Python
  https://www.python.org/downloads/
- Install PIP
  https://pip.pypa.io

pip install <module>

- virtualenv
- pip install flask==0.10.1

About Me

Senior Information Security Engineer

Cisco Security & Trust Organization

Research Triangle Park, North Carolina

Certified Information Systems Security Professional

```
SEC-USR-MESS-2 SEC-RUN-SAFEC SEC-RUN-OSC SEC-RUN-ASLR SEC-PWD-MAXLIFE SEC-PWD-AUDIT SEC-OPS-RDY
 SEC-OPS-NPROD SEC-NRCV-CRED-5 SEC-LOG-TENISOL-2 SEC-LOG-STATCHG SEC-LOG-LOGCONF SEC-LOG-LIMIT
SEC-LOG-CONTENT SEC-LOG-ATTACK SEC-LOG-ADMIN SEC-IPS-ESP-5 SEC-INSEC-USER SEC-CSP-LOGTRUST SEC-
CRY-LOG SEC-AUT-LOGZCHG SEC-AUT-LOGNCHG SEC-WEB-HTTPMETH-2 SEC-SRT-311000-3 SEC-OPS-BUSCONT-2 S
EC-LOG-CENTRAL-3 SEC-INF-HARDEN-2 SEC-509-FQDN-2 SEC-SET-TIME SEC-SRT-311000-2 SEC-LOG-INDC-2 S
EC-SRT-DTLS-2 SEC-CHG-LOGD-3 SEC-CRE-NOLOCK SEC-IP-IPv6 SEC-AUD-FIELD-3 SEC-OPS-BUSCONT SEC-INF
-DOS SEC-INF-AVSCAN SEC-OPS-REVOKE SEC-DAT-RETAIN SEC-DAT-REMOVE SEC-DAT-AUDIT SEC-WEB-NOREDIR
SEC-WEB-IDCACHE SEC-WEB-IDCTX SEC-SDP-311030 SEC-SRT-311020 SEC-SRT-311010 SEC-TIME-NTP SEC-ASU
-TRAIN-2 SEC-WEB-SMTIME-2 SEC-WEB-STATE SEC-WEB-CRYPCOOK-2 SEC-WEB-RESP-2 SEC-WEB-SESCOOK-2 SEC
-OPS-MAXLIFE SEC-DAT-VMCRYPT SEC-WEB-IDATTR-2 SEC-AUT-CREROT SEC-LOG-CENTRAL-2 SEC-AUT-LOG SEC-
LOG-RET SEC-WEB-NPCOOKIE-2 SEC-DAT-SANITIZE SEC-DAT-KEYMGMT SEC-ALL-LOGD-2 SEC-SW-NEEDONLY SEC-
ASU-STATIC SEC-HTP-SSL3-3 SEC-ASU-TMOD SEC-OPS-LEGAL SEC-OPS-INCPOL SEC-INF-MGTZONE SEC-CRE-PRI
V SEC-CRE-MULTIFAC SEC-DAT-GEOG SEC-DAT-SEGR SEC-DAT-BACKUP SEC-DAT-MEDIA SEC-WEB-CLCKJACK SEC-
WEB-IDVALID SEC-PRV-KNOWWHAT SEC-PRV-NOCOLL SEC-PRV-USERAUTH SEC-PRV-ERASE SEC-DAT-PRIV-2 SEC-P
RV-LOCALID SEC-PRV-MANAGE SEC-TLS-CURR-3 SEC-LOG-FORM SEC-OPS-RESTCRYP-2 SEC-WEB-XSSUBCXT-2 SEC
-WEB-HTTPMETH SEC-WEB-SQLIN-2 SEC-509-CHAIN SEC-DAT-BACKUPTS SEC-LOG-TRANS SEC-DAT-SOP SEC-WEB-
URLPARM SEC-WEB-SYSIN-2 SEC-WEB-ACCBYURL-2 SEC-LOG-PROTO SEC-LOG-APPREQ SEC-WEB-IDALL-2 SEC-LOG
-ACCESS-2 SEC-WEB-CSRF SEC-DAT-BACKUPER SEC-DAT-BACKUPI SEC-509-LIFETIME SEC-509-CERTEXT SEC-CR
E-LIMTRY-2 SEC-LOG-NOSENS-2 SEC-WEB-INJIN-2 SEC-WEB-SANIHTML SEC-WEB-AUTOCOMP SEC-DAT-BACKUPET
SEC-LOG-INFRAREQ SEC-WEB-ID-2 SEC-DAT-ISMS-2 SEC-WEB-XSS SEC-509-FQDN SEC-HTP-HSTS SEC-DAT-MONE
Y SEC-WEB-XPATHIN-2 SEC-509-REVOKE SEC-SUP-PATCH SEC-UPS-REGI SEC-INT-CRED-2 SEC-DEF-CRED-2 SEC
-OPS-CHGMGT SEC-WEB-SCAN SEC-INF-SCAN SEC-INF-THIRD SEC-INF-DEVNET SEC-INF-ZONES SEC-INF-MONITO
R SEC-INF-FWSEP SEC-INF-VER SEC-INF-HARDEN SEC-OPS-ASTMGT-2 SEC-OUT-CRED-2 SEC-OPS-STRENGTH SEC
-CRY-PRIM SEC-OPS-SEG-2 SEC-AUT-ACCDEF SEC-CRY-STDCODE SEC-CRE-SHARE SEC-OPS-SUDO SEC-AUT-AUTH
SEC-OPS-PUBCRYP-2
```


WEATHER
AUTHORITY
TEMP
DPT 58°
MAX

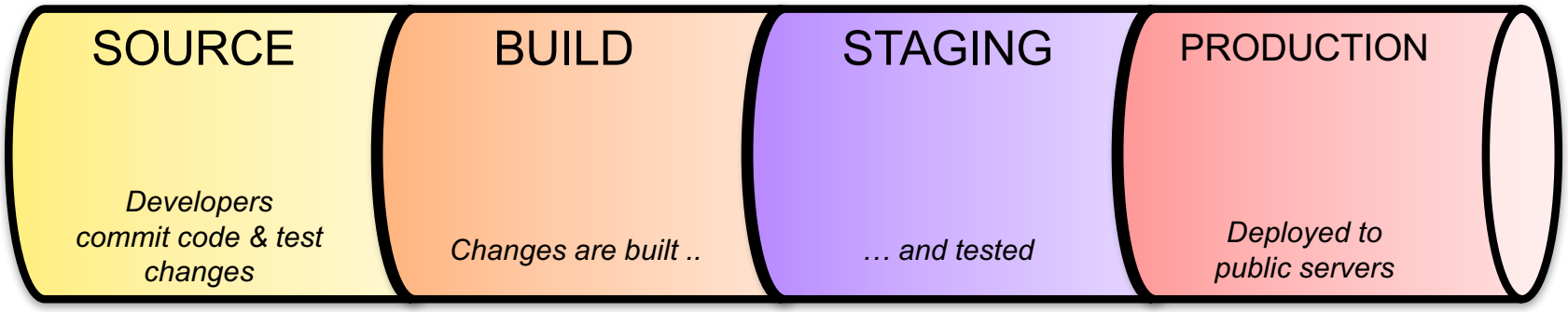

Triangle
InfoSeCon
triangleinfosecon.com
Friday October 27, 2017
Raleigh Convention Center, Raleigh, NC

In this workshop we will setup

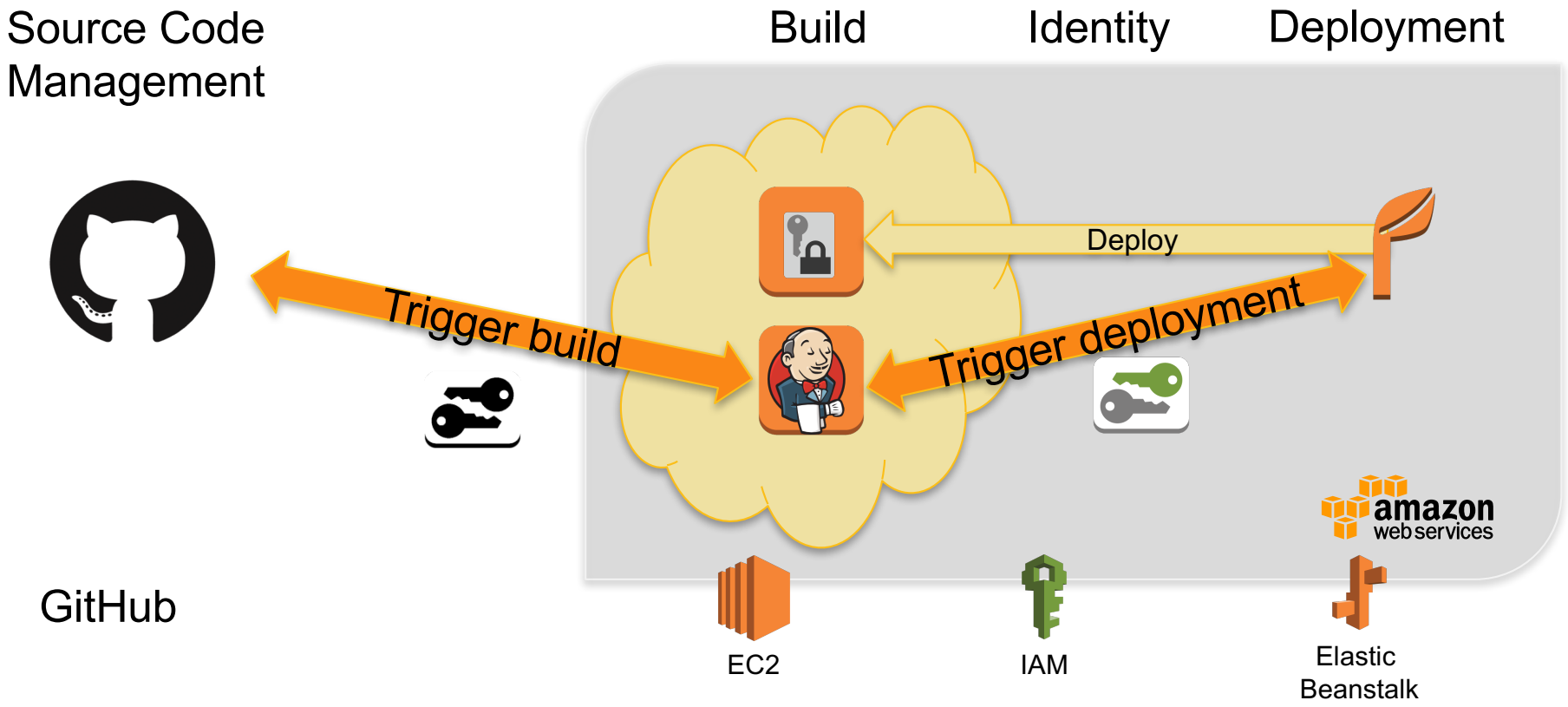
- A source code repository
- A centralized build server
- Deploy a simple micro service into the cloud

Along the way, we'll form them into a continuous integration toolchain … *securely*

Code pipeline



Our Continuous Delivery Pipeline



Agenda

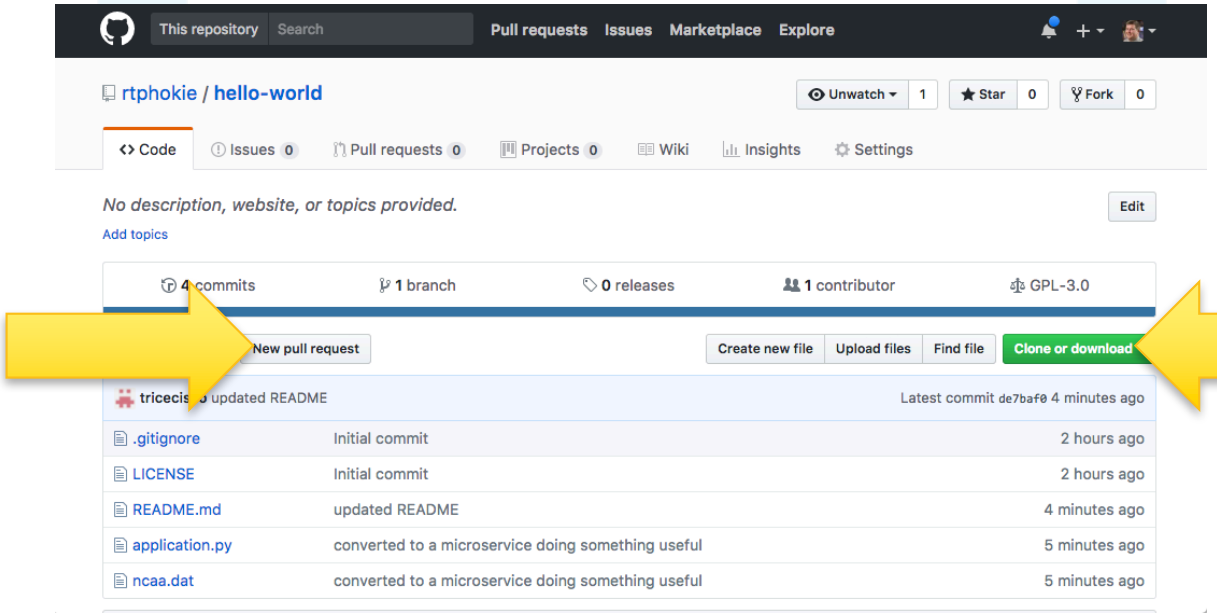
- 1-1:30 overview / AWS and Git prerequisites
- 1:30-1:45 Fork the GitHub Hello Word Repo
- 1:45-2:30 Provision EC2, Elastic Beanstalk and security groups
- 2:30-3:00 install Jenkins with Git and EB plugins
- break
- 3:30-4:30
  - trigger Jenkins builds from GitHub updates via keys,
  - trigger deployment from Jenkins to Elastic Beanstalk via keys
- 4:30-5:00 CI/CD best practices

Cisco Public 2017 All Rights Reserved

Get the app source code from GitHub

- Visit `https://github.com/rtphokie/hello-world`

Pull …
then clone



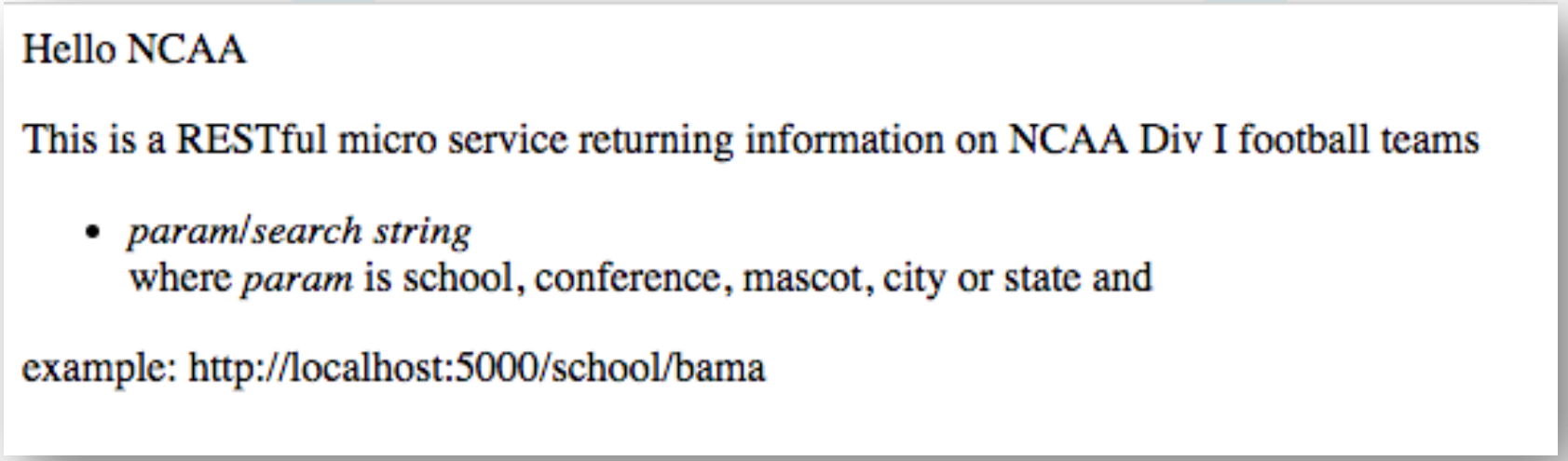
Clone

Get the app running locally

1. `virtualenv ~/eb-virt`
2. `source ~/eb-virt/bin/activate`
3. `git https://github.com/rtphokie/hello-world.git` or download from Box (`http://bit.ly/isaca-hello-world`)
4. `cd hello-world`
5. `pip install -r requirements`

Test application locally

1. `python application.py`
2. Visit `http://localhost:5000`


Hello NCAA

This is a RESTful micro service returning information on NCAA Div I football teams

- *param/search string*
  where *param* is school, conference, mascot, city or state and

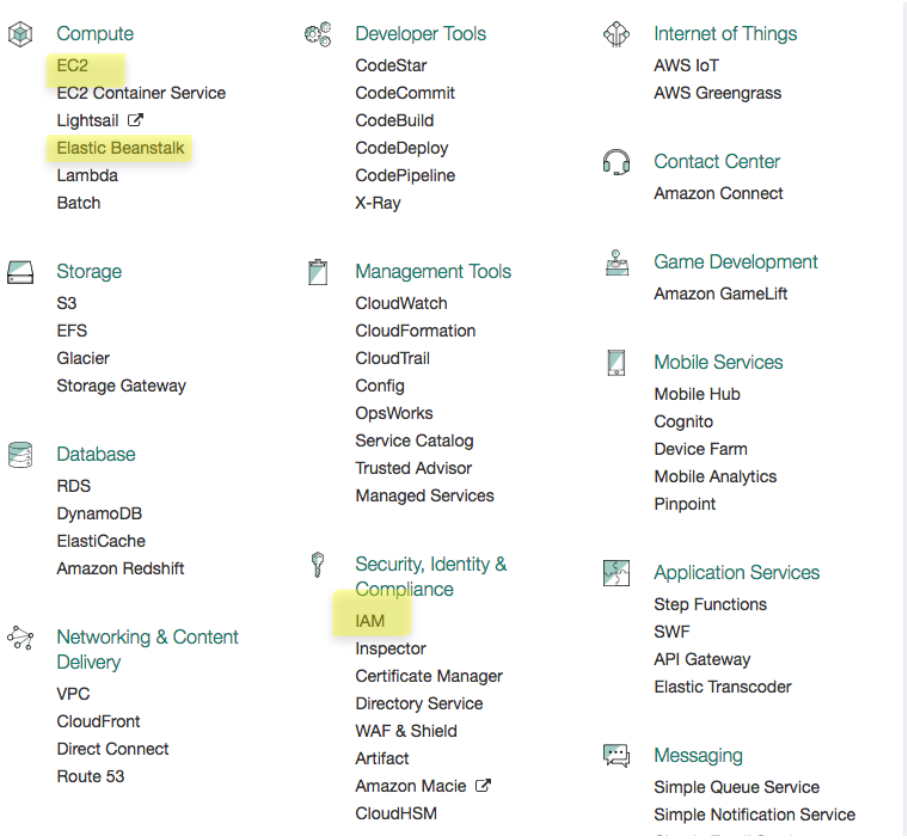
example: http://localhost:5000/school/bama

Our Continuous Delivery Pipeline

Source Code
Management

GitHub

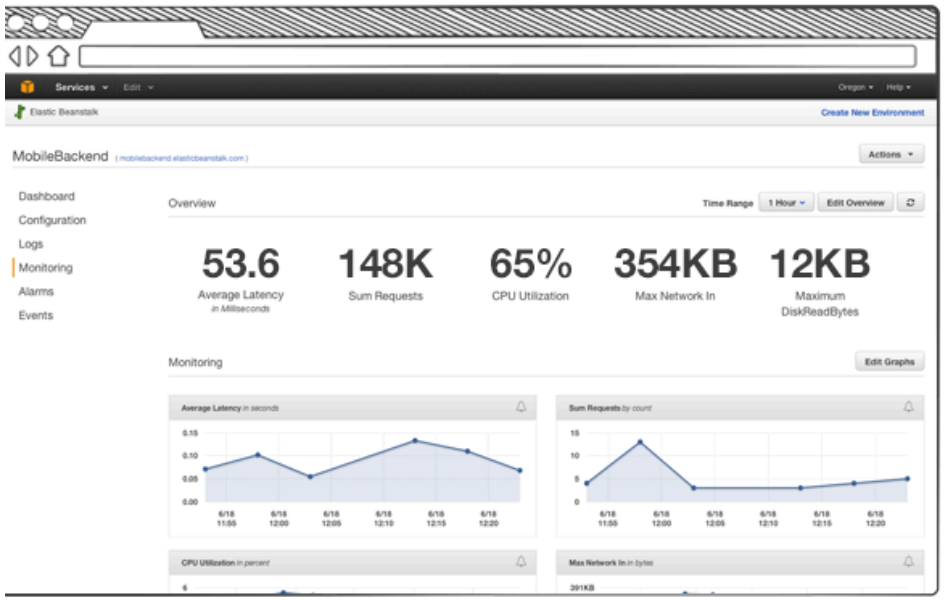
AWS console



We will be using

- EC2
  - Security Groups
  - Instances
- IAM
- Elastic Beanstalk

Elastic Beanstalk: Provision



Welcome to AWS Elastic Beanstalk

With Elastic Beanstalk, you can **deploy**, **monitor**, and **scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an application source bundle and then create a new application. If you're using **Git** and would prefer to use it with our command line tool, please see Getting Started with the EB CLI.

To deploy a **sample app** with just one click, se tform and click **Launch Now**.

By launching the s plication, you allow A ic Beanstalk to administer AWS resources and necessa ns on your behalf. L

Python king for a different platform? Let us know.

AWS Elastic Beanstalk will create a nment running Python 3.4 on 64bit Amazon Linux 2017.03 v2.5.2. Change platform version.

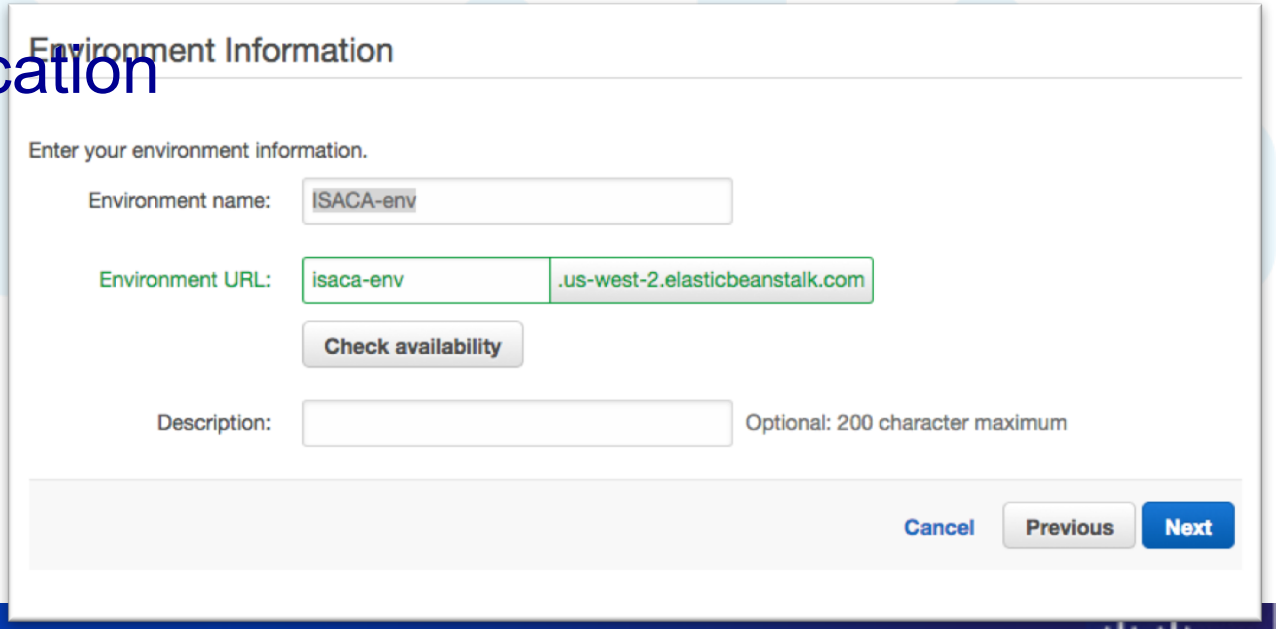
Launch Now

2.7 on 64 Bit Linux

cisco

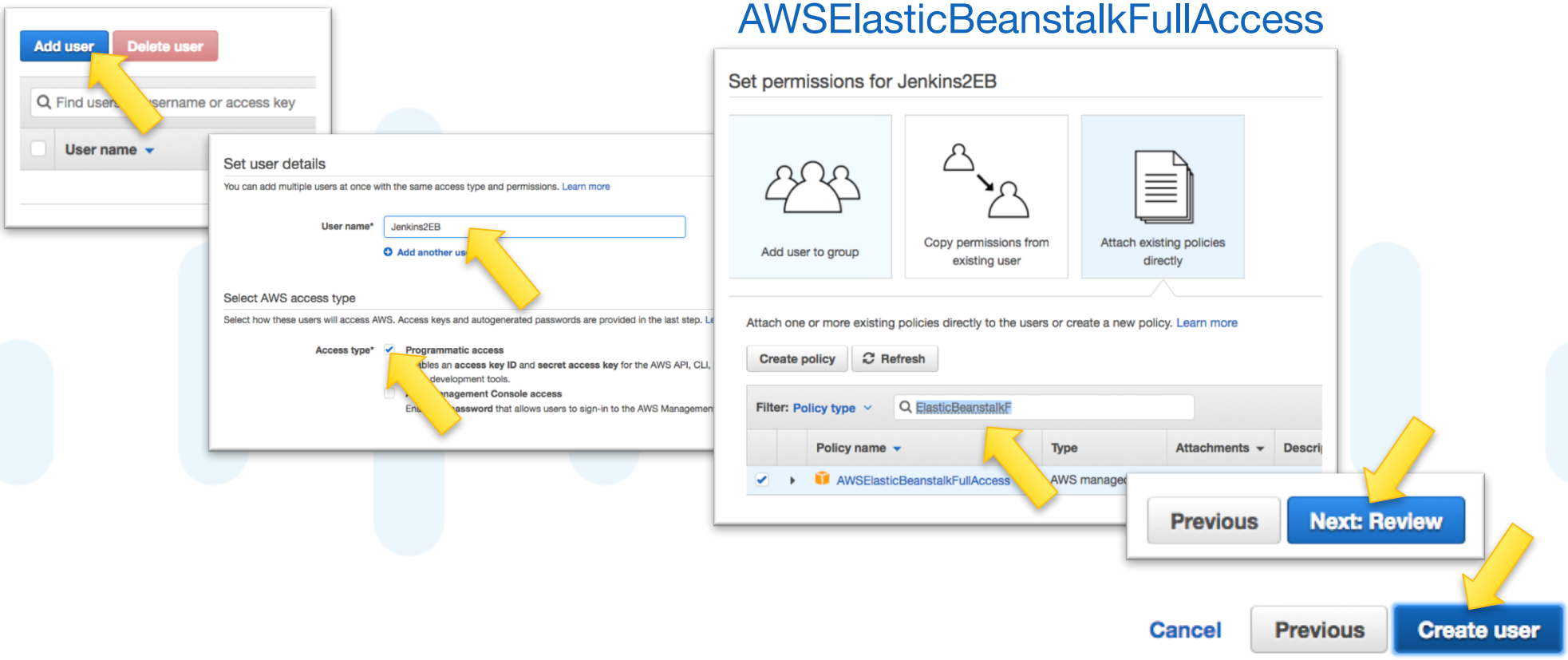
Elastic Beanstalk provision

- Create Environment
  - Create Webserver (Python, change to 2.7; single instance)
  - Use sample application
  - Launch



IAM: Provision EB User

AWSElasticBeanstalkFullAccess



Our Continuous Delivery Pipeline

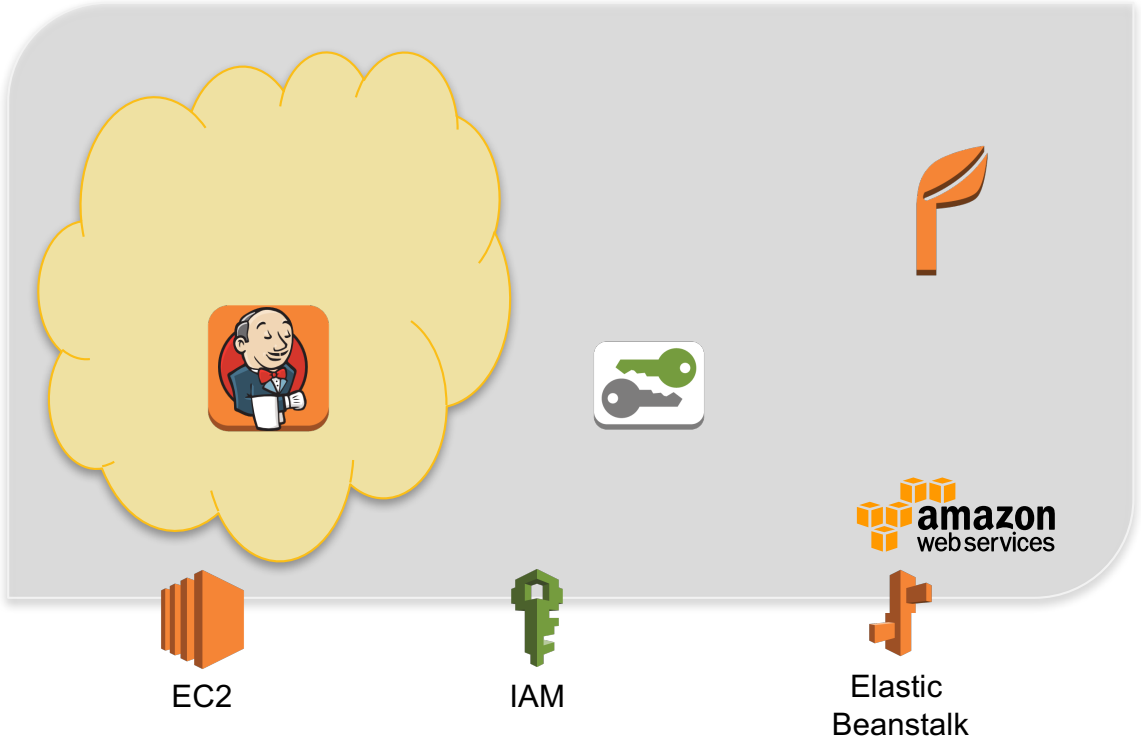
Source Code Management

GitHub

Build

Identity

Deployment

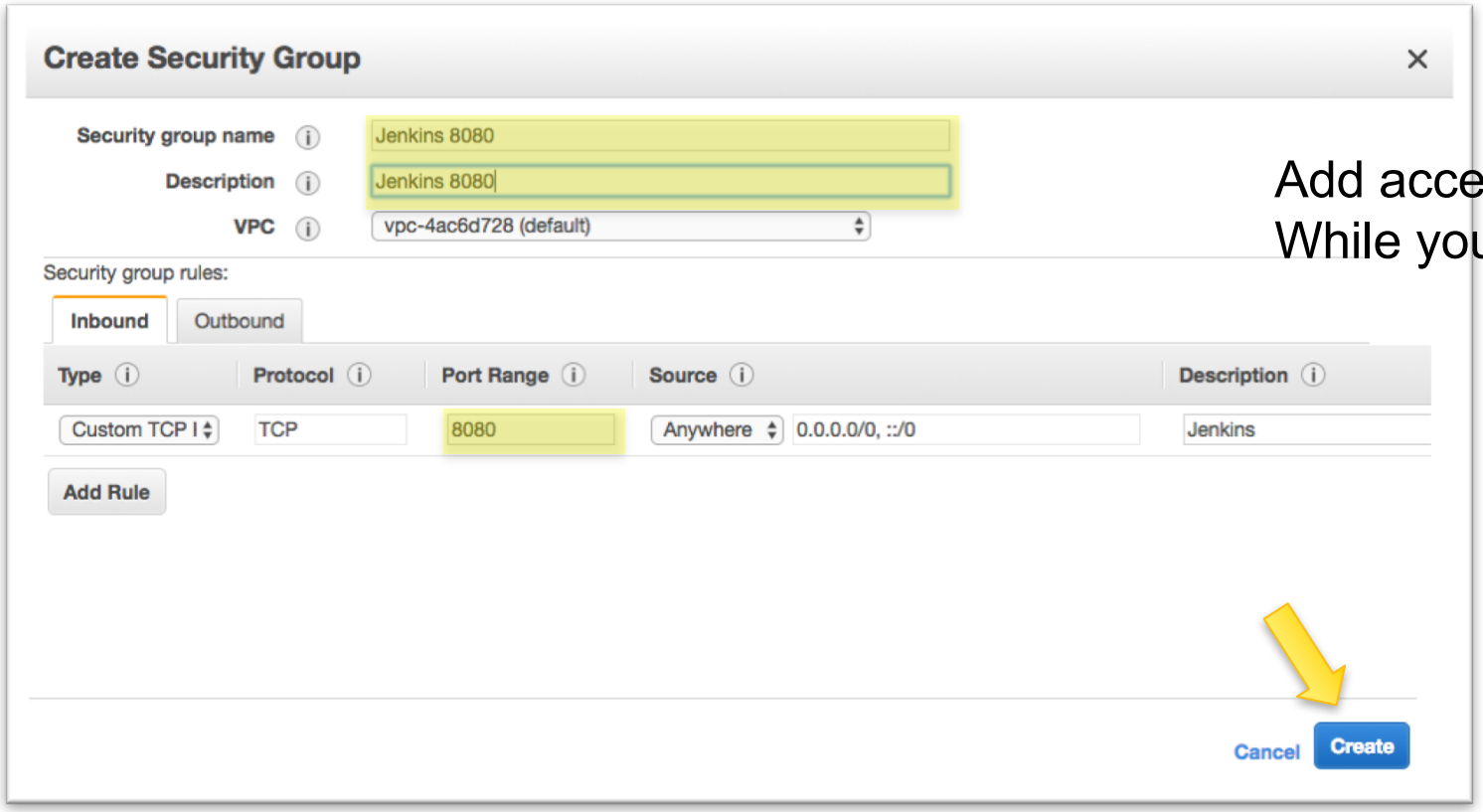


EC2

IAM

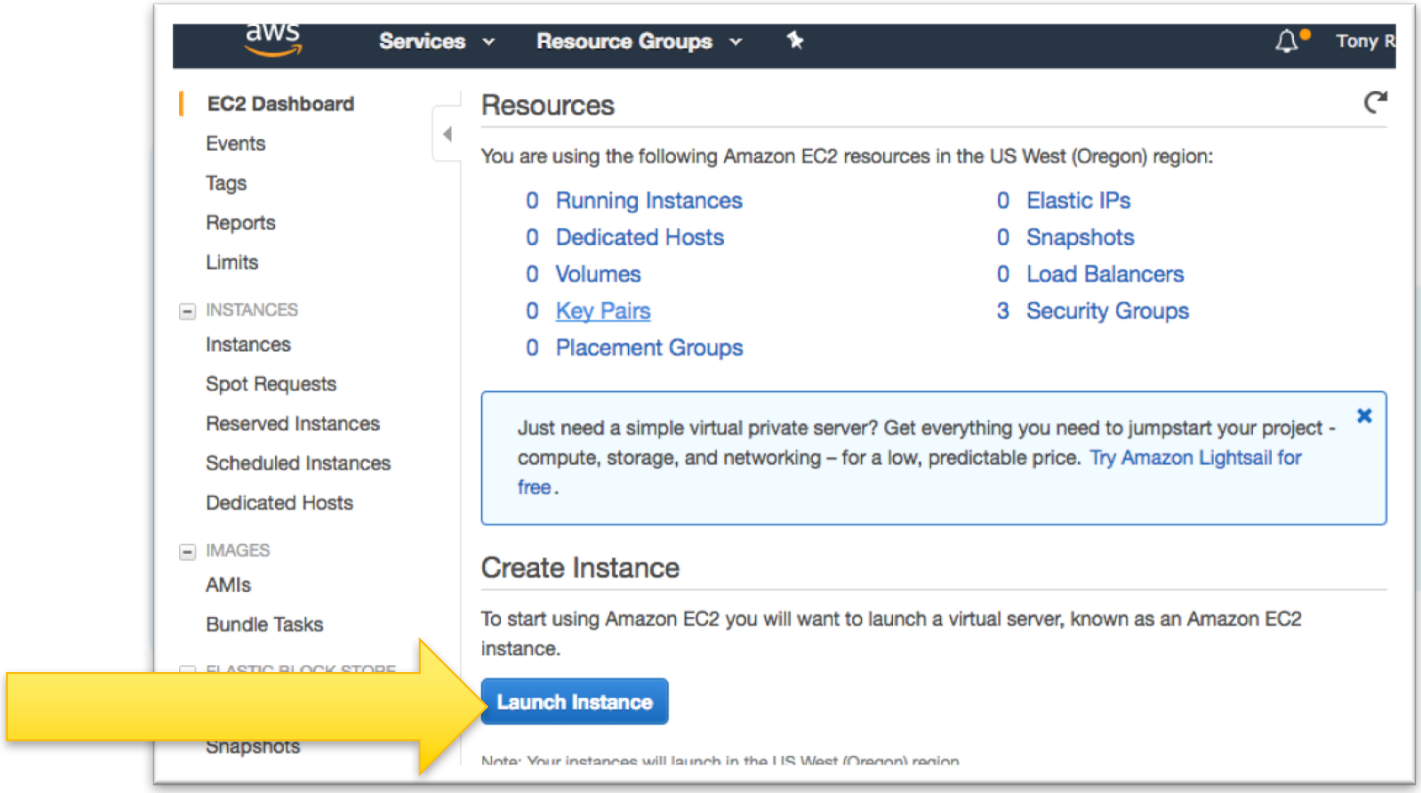
Elastic Beanstalk

EC2: security group

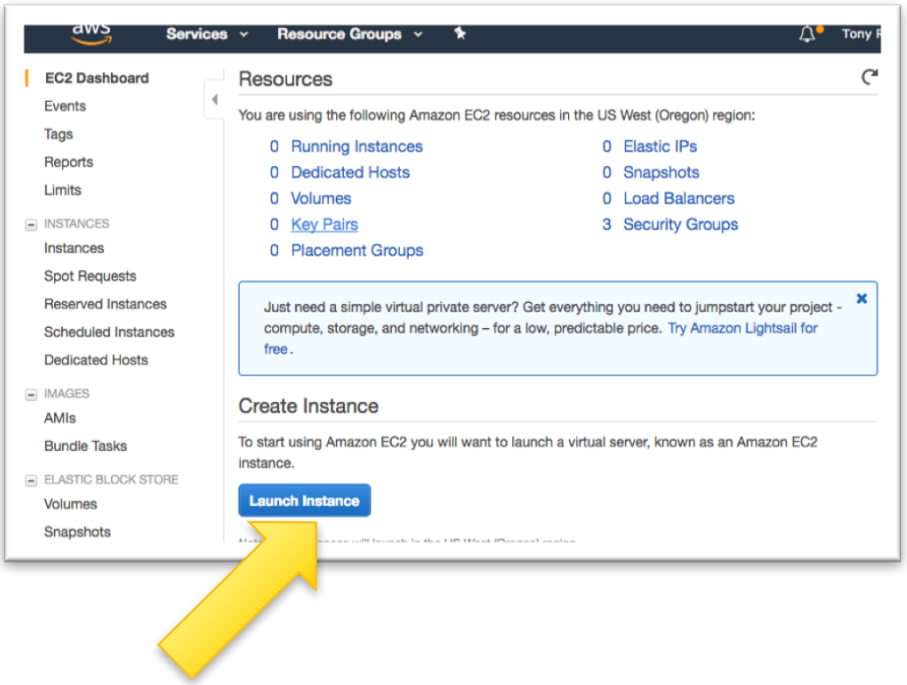


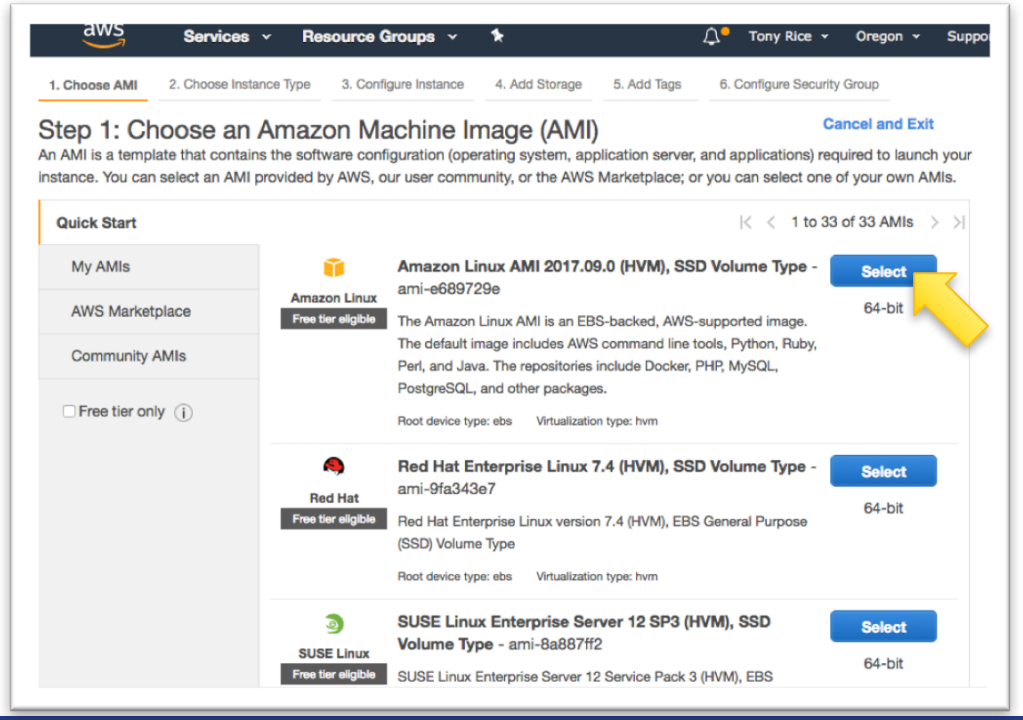
Add access to port 22
While you are creating this

EC2: launch EC2 instance

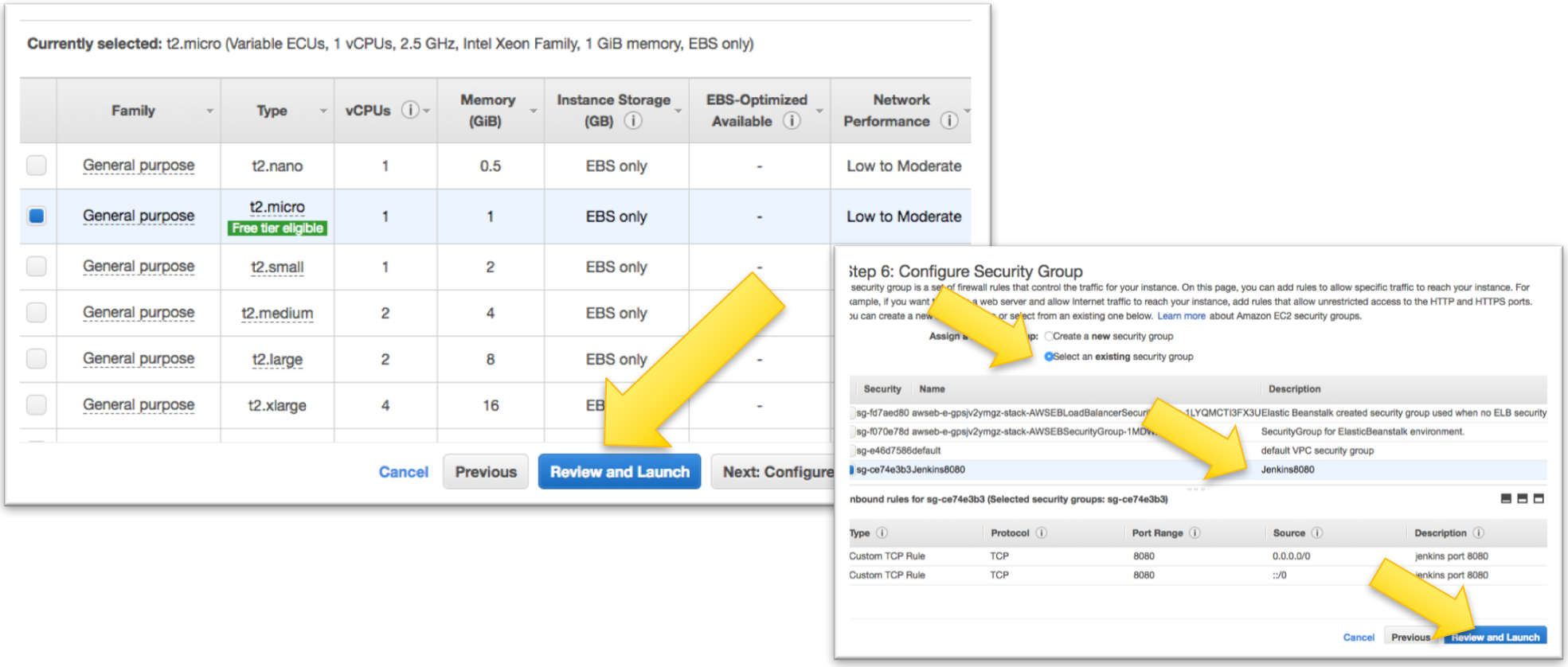


AWS: launch EC2 instance





Select a free tier eligible instance type, launch



Create key pair & launch

~/.ssh/config shortcut

```
host isaca
    User ec2-user
    Hostname 34.215.204.5
    IdentityFile /Users/trice/Downloads/ISACA_workshop.pem
```

```
$ sudo yum –y update
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-
main
                                                                | 2.1 kB  00:00:00
amzn-
updates
                                                                | 2.5 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package amazon-ssm-agent.x86_64 0:2.1.4.0-1.amzn1 will be updated
```

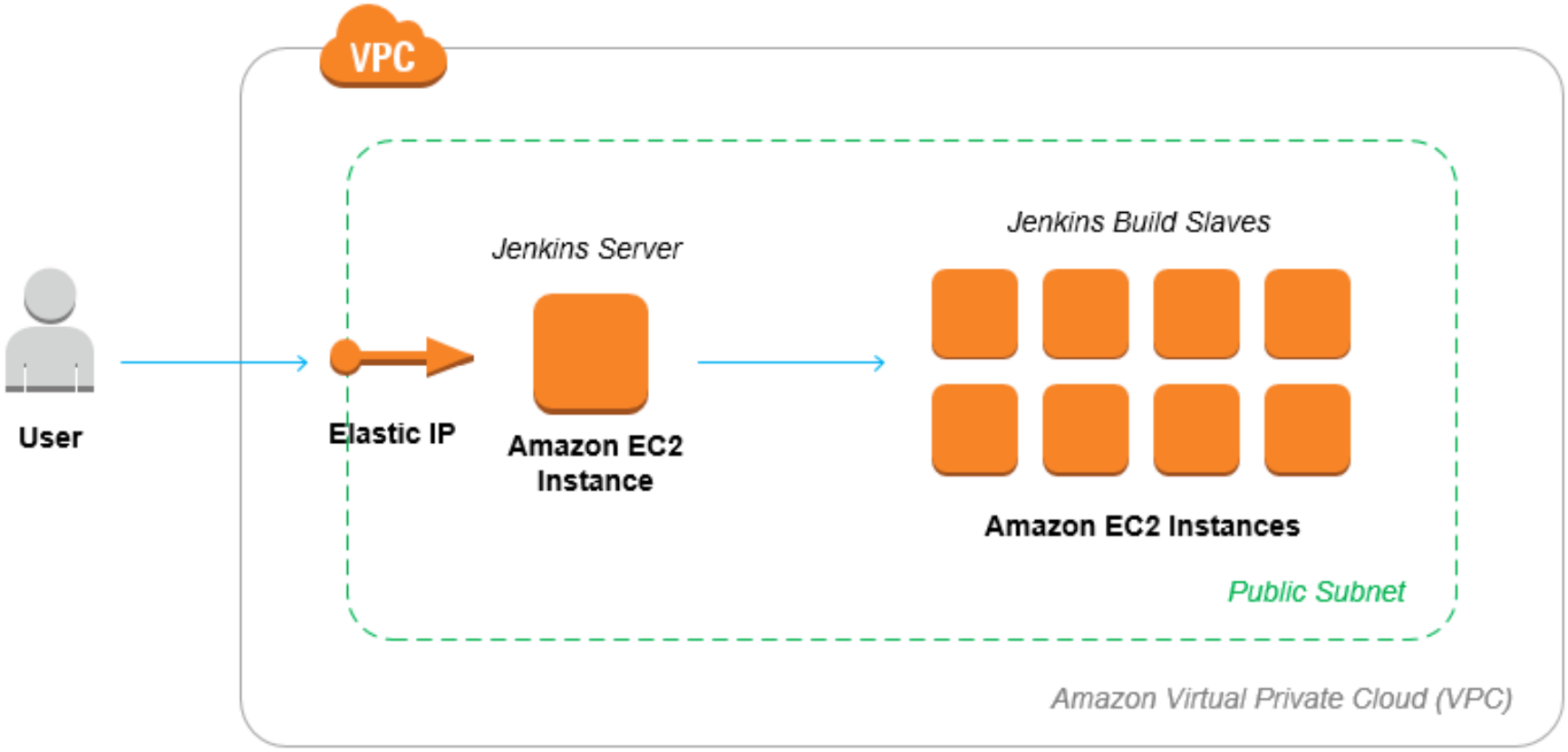
Login!

```
$ chmod 600 ~/Downloads/ISACA_workshop.pem
$ ssh -i ~/Downloads/ISACA_workshop.pem ec2-user@34.215.204.5
Warning: No xauth data; using fake authentication data for X11
forwarding.
X11 forwarding request failed on channel 0


       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|


https://aws.amazon.com/amazon-linux-ami/2017.09-release-notes/
No packages needed for security; 5 packages available
Run "sudo yum update" to apply all updates.
```

Jenkins



Jenkins: installation

```
$ sudo yum update –y
$ sudo yum install -y git

$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
   http://pkg.jenkins-ci.org/redhat/jenkins.repo
$ sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key
$ sudo yum install –y jenkins

$ sudo service jenkins start
Starting Jenkins                                    [  OK  ]
```

If you get Java8 errors

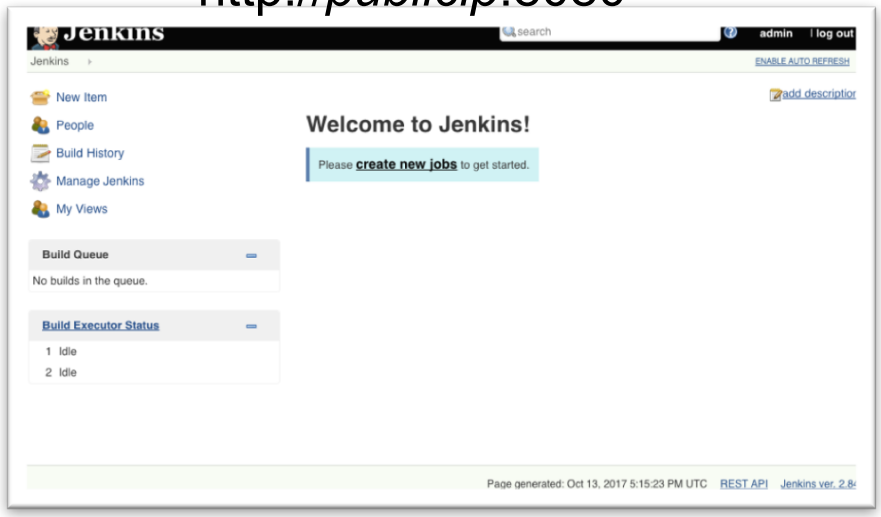
```
$ sudo service jenkins start
Starting Jenkins Jenkins requires Java8 or later, but you are running 1.7.0_151-
mockbuild_2017_08_09_21_42-b00 from /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.151.x86_64/jre
java.lang.UnsupportedClassVersionError: 51.0
at Main.main(Main.java:124)
$ sudo yum –y install java-1.8.0
$ sudo yum –y remove java-1.7.0-openjdk
$ sudo service jenkins start
Starting Jenkins                                                    [   OK   ]
$
```

Jenkins: Login

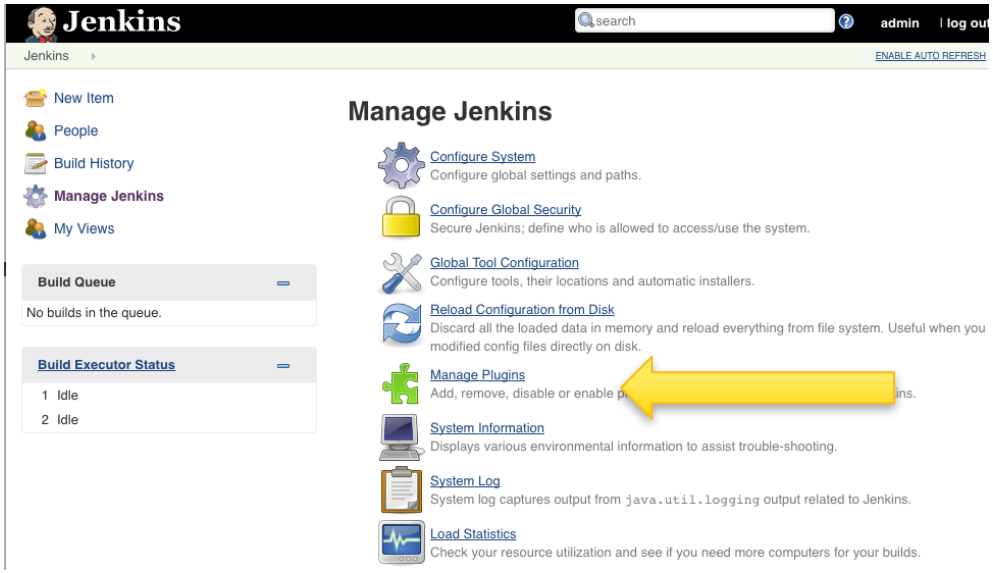
From EC2 Linux prompt

```
$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
somestring
```

http://*publicip*:8080



Jenkins: Plugins



Install these plugins

- AWS Elastic Beanstalk Deployment Plugin (not Publisher)
- GitHub Authentication

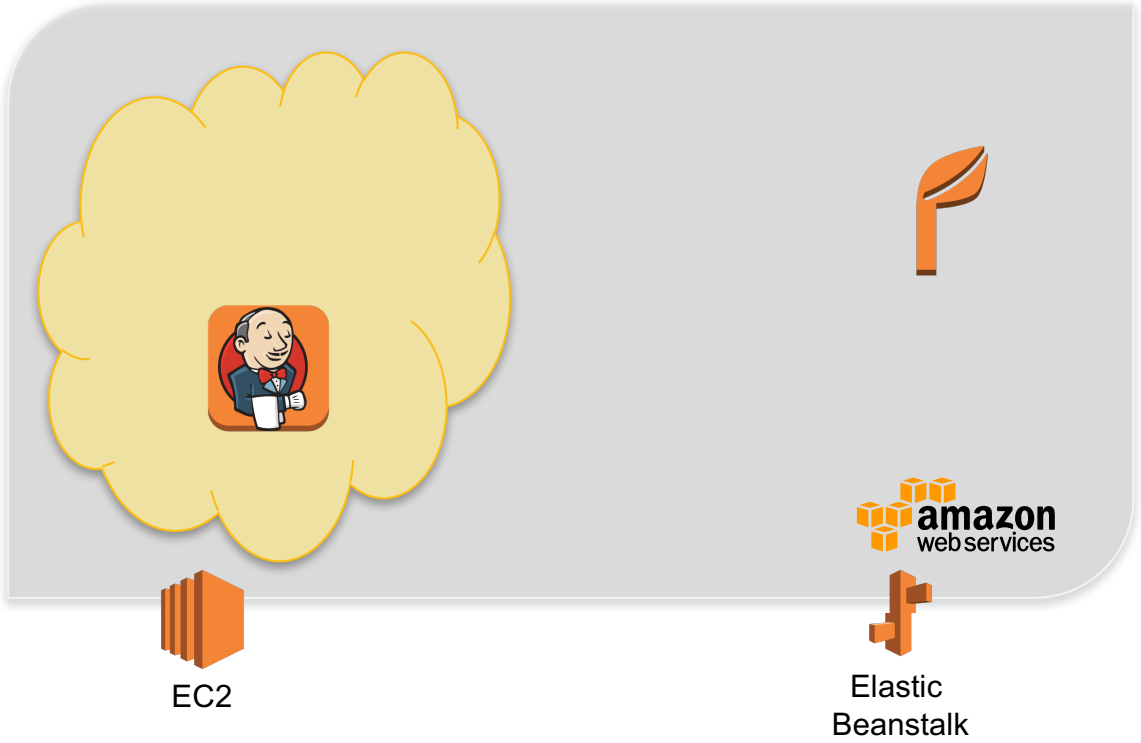
Our Continuous Delivery Pipeline

Source Code Management

Build

Identity

Deployment



GitHub

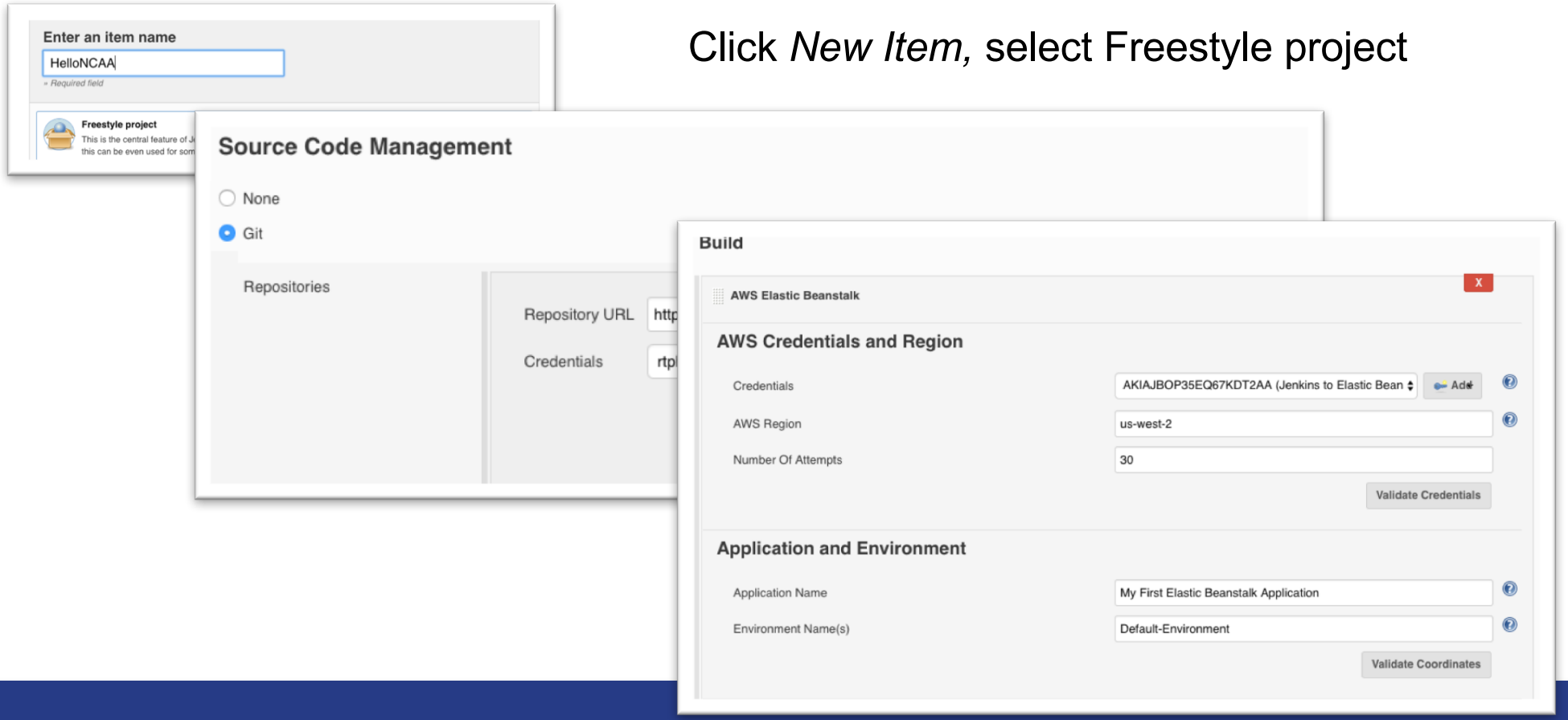
EC2

Elastic Beanstalk

Create a key pair

- Login to EC2 instance
- sudo su – jenkins
- ssh-keygen -t rsa
- Public and private is_rsa files are in ~jenkins/.ssh

Create a build job



Click *New Item,* select Freestyle project

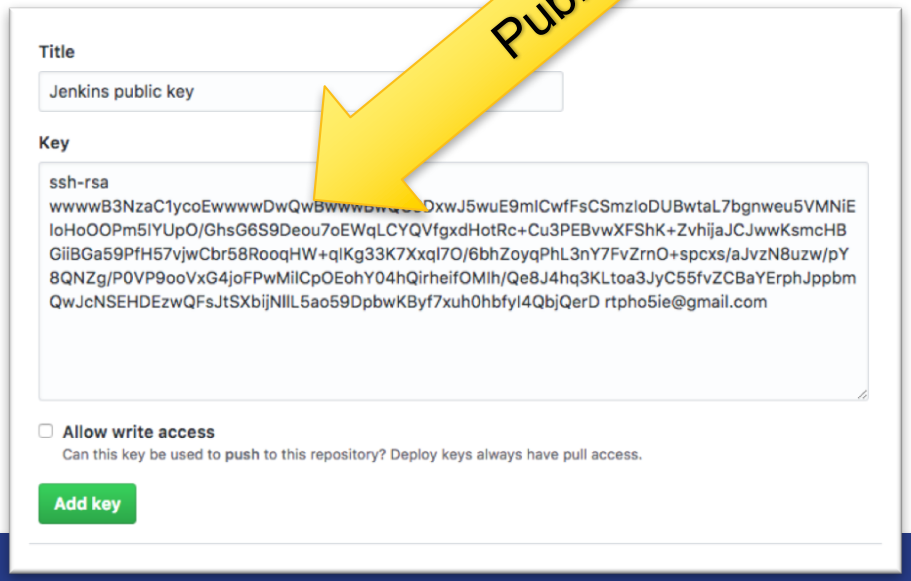
Github: Deploy key

```
$ ssh-keygen -t rsa -C 'trice@cisco.com'
```

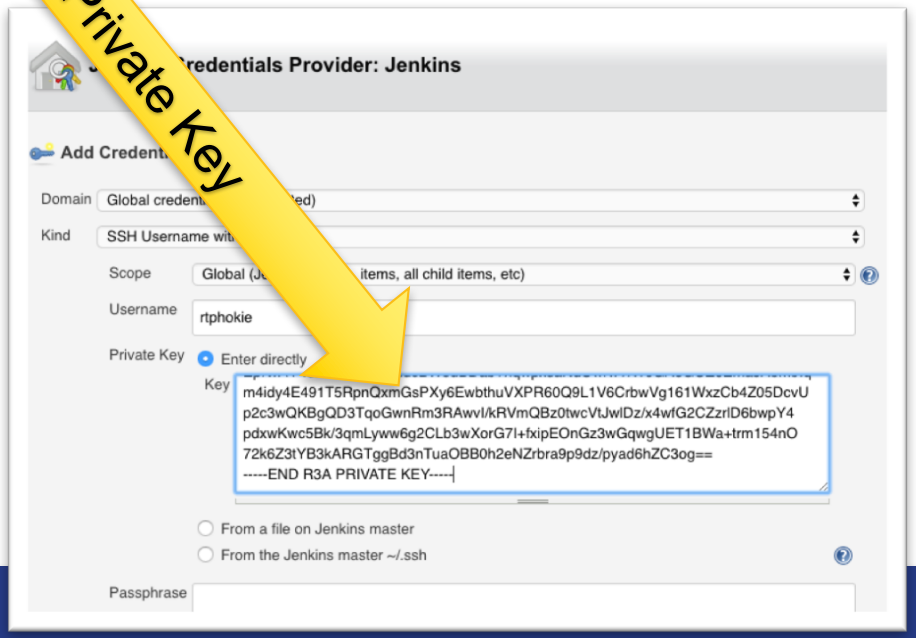
Public key

Private Key

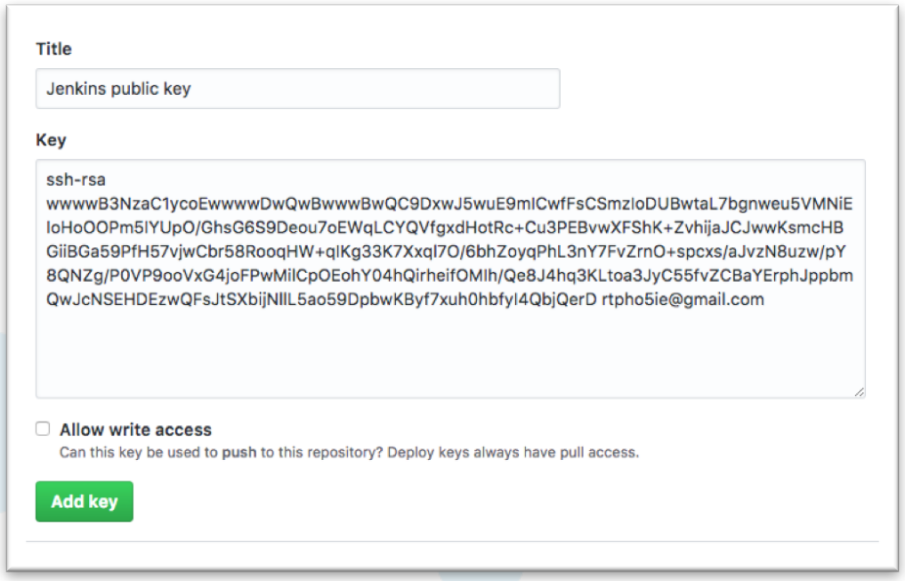
Github



Jenkins



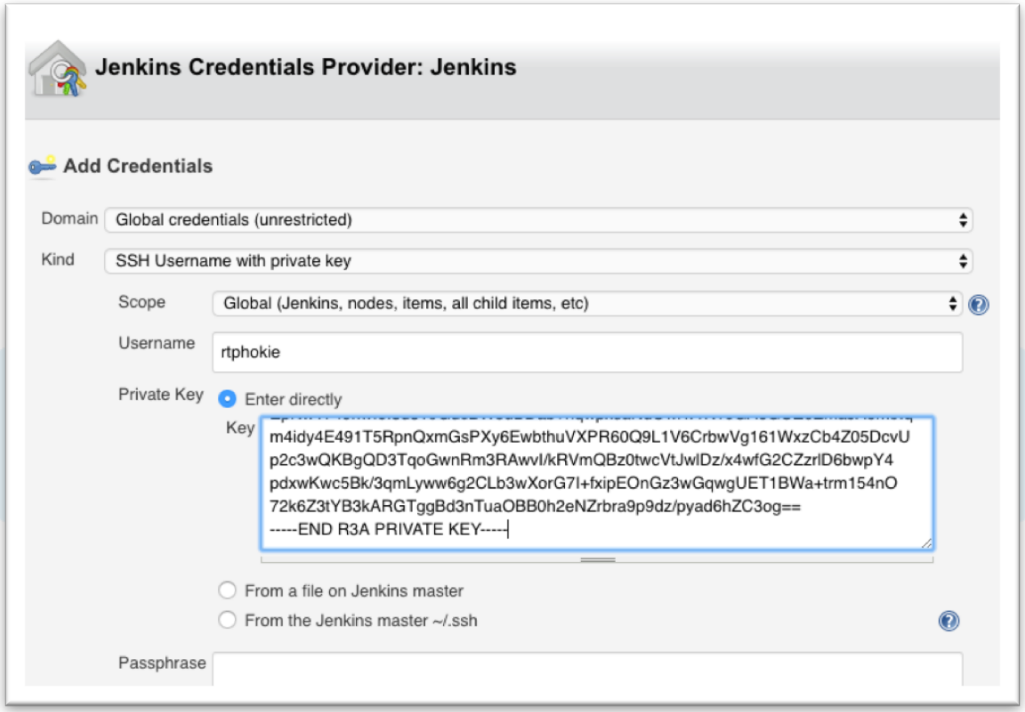
Apply Public Key to Github



- Settings >> Deploy keys
- Click Add Deploy Key

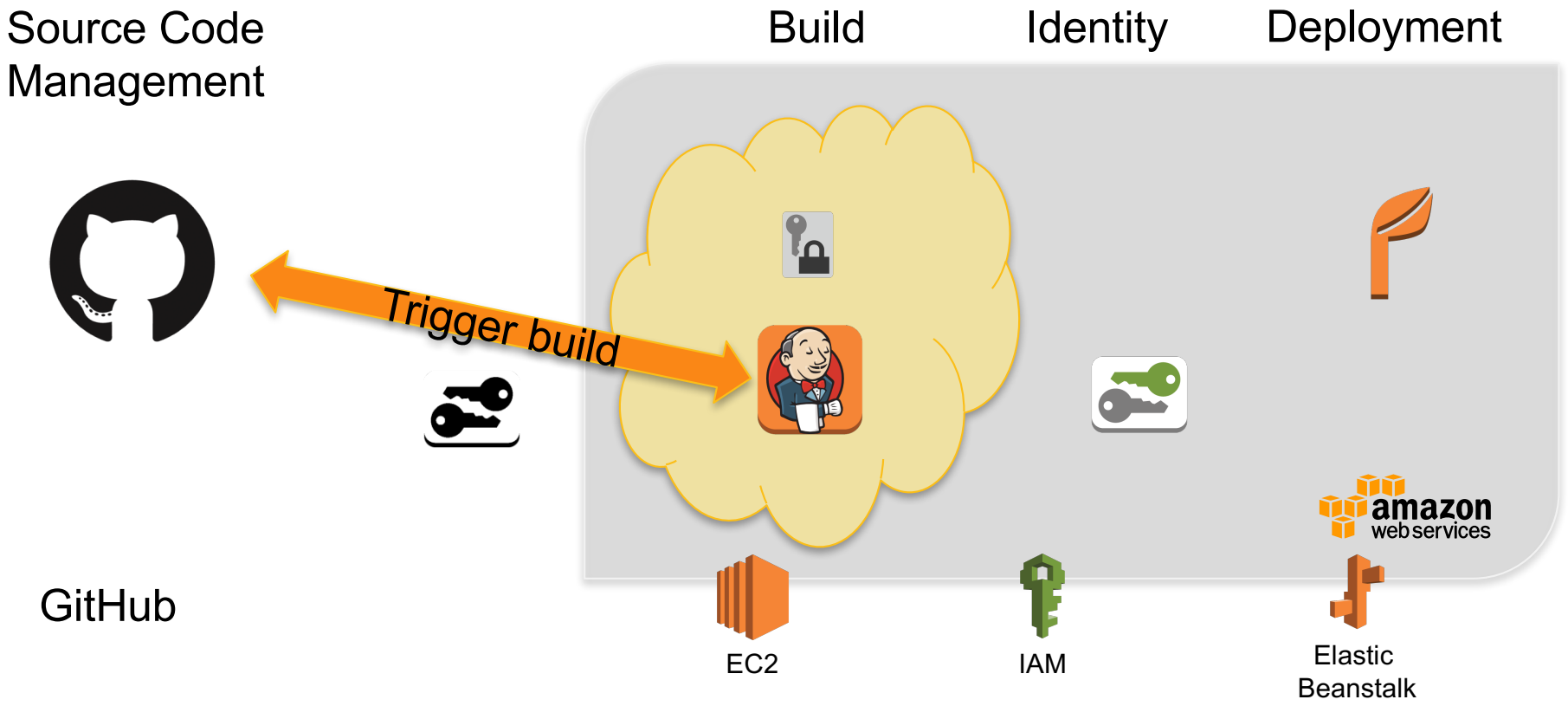
```
enter public key
enter a title
```

Apply Private Key to Jenkins Job

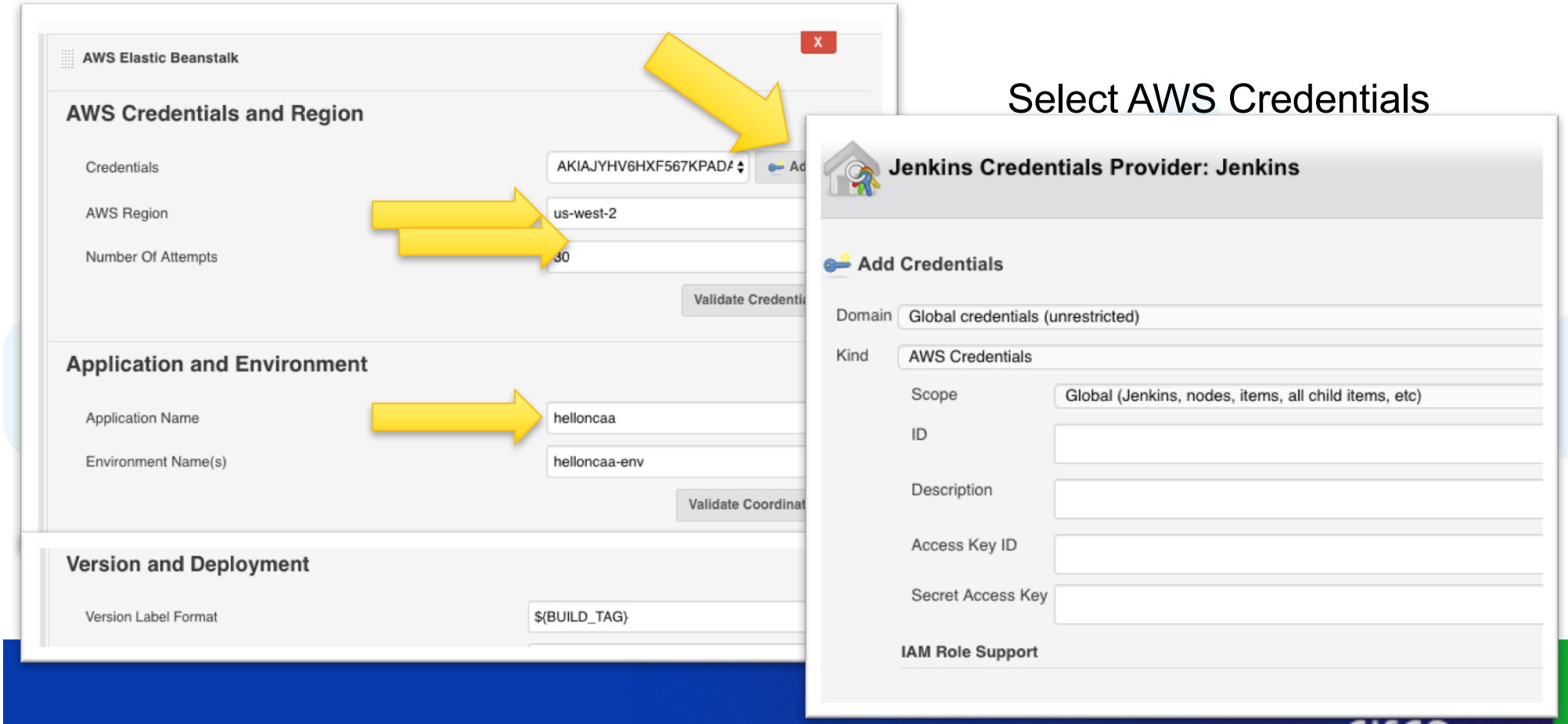


- New Freestyle Job
- Source Code Mgmt->Git
  url:
  `https://github.com/rtphokie/hello-world.git`
- `Click add key`
  `SSH with key`
  `enter userid`
  `enter Private key`
  `click add`

Our Continuous Delivery Pipeline

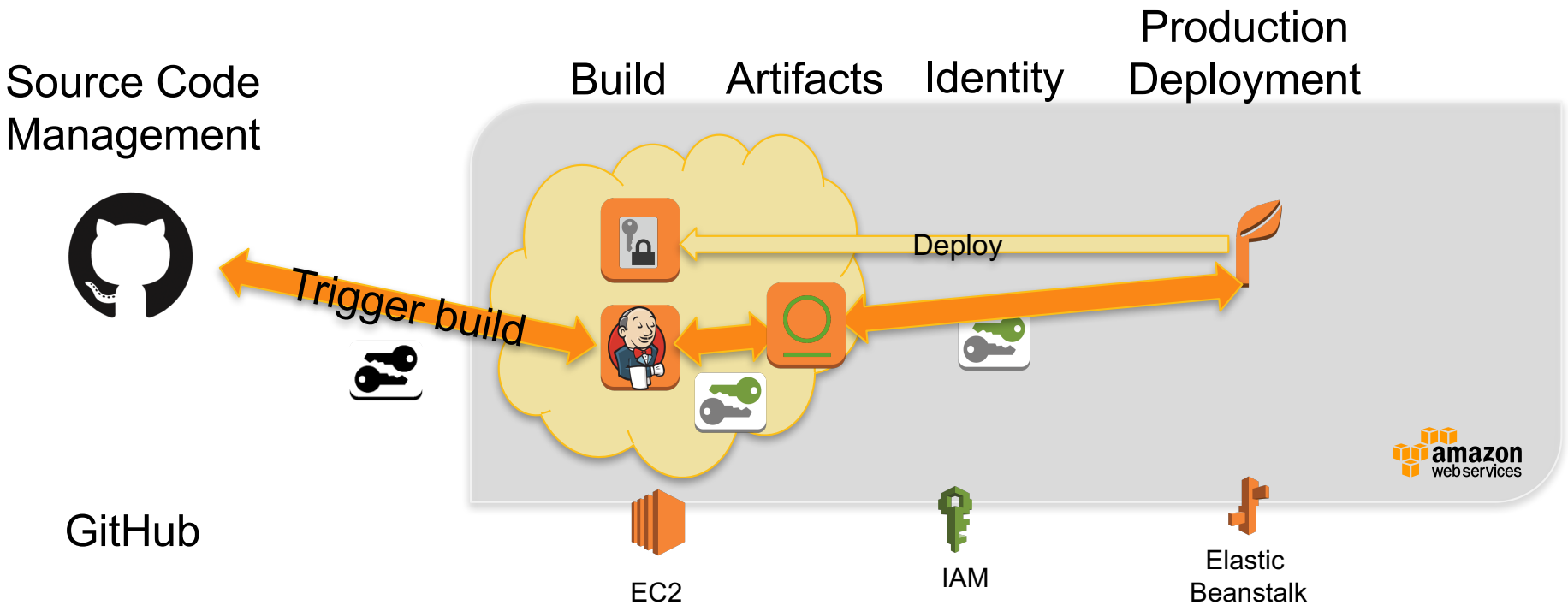


Jenkins: deploy to EB post build

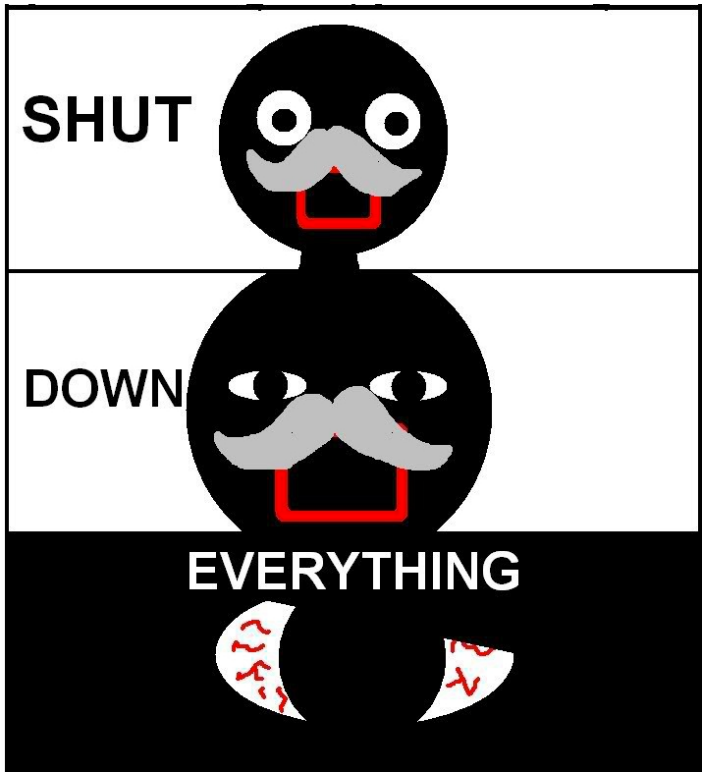


CISCO

Artifact Management



Before you go



AWS bills based on running and stopped instances. To avoid unexpected bills:

1. Shutdown your EC2 instances and Elastic Beanstalk apps.
2. Terminate them ASAP

Thank you.

trice@cisco.com

@rtphokie