Public Clouds: AWS EC2

Outline

- What is AWS and EC2
- Sign-up EC2 for a free Account
- Creating a instance with a pre-packaged AMI
- Create a custom AMI

What is AWS

- What to do with over-sized Amazon data centers out of holiday season?
  - By the end of 2003 Chris Pinkham and Benjamin Black proposed to provide some sort of Queuing System to use the machines in something-else

- In 2007 AWS start to provide online services to other providers or client-side applications

- Today provides a wide range of services
  - Compute
    - EC2 (Elastic Cloud 2), EMR (Elastic Map Reduce)
  - Networking
    - Route 53 (DNS service), Virtual Private Cloud (VPC) [Isolated EC2], AWS direct connect
  - Storage
    - S3, Amazon Glacier (Backup), AWS Storage Gateway (block level device through iSCSI), EBS (Elastic Block Store == LVM)
  - Data base
    - DynamoDB (NoSQL), ElasticCache (memcached), RelationalDB (Oravle, Mysql…)
  - Many SaaS services

What is EC2?

- IaaS to other AWS services but also available for external customers
  - "virtual" computers that you configure-as-you-want and pay-as-you-go
- Different instances according node characteristics and on-demand costs
  - m1.small --- 2 GB Mem, 1 vCPU, 160GB HDD, $0.07 c/hour
  - ….
  - hs1.8xlarge – 117GB, 16 vCPU, 48TB, 10Gbit net, $4.6 c/hour
- Class of instances
  - cX (compute), rX (memory), hsX (storage), gX (GPGPU)
  - Prices are data-center dependent
- Persistent storage
  - Uses Elastic Block Storage
- Persistent IP
  - Uses elastic IP
- Provides a free tier to learn!
  - **http://aws.amazon.com/es/free/**

EC2 Virtual Data Center

- Provides the resources to deploy a “complex” data center without spend a “penny” before to start “earning” money: i.e. Pay-as-you-go

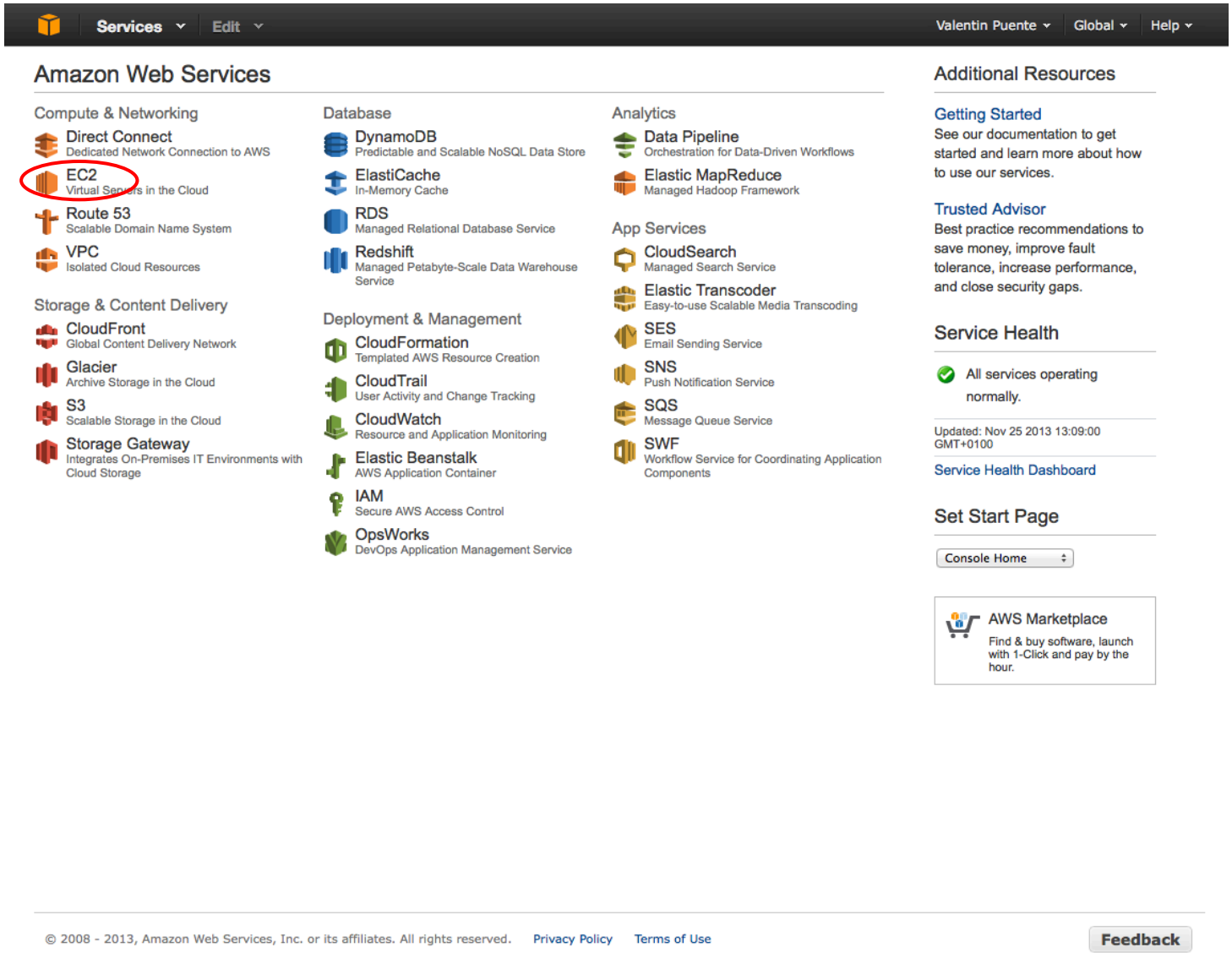
- Elements in a data-center (AWS-Terminology)
  - Servers (Instances)
  - Static IP Addresses (Elastic IP Addresses)
  - Firewall (“Security Group”)
  - SAN (“Elastic Block Store”)
  - Monitoring (“CloudWatch”)
  - Load Balancing (“Elastic Load Balancing”)

- Key: auto-scaling
  - **Up**: To meet demands
  - **Down**: To save money
  - The changes are produced in minutes

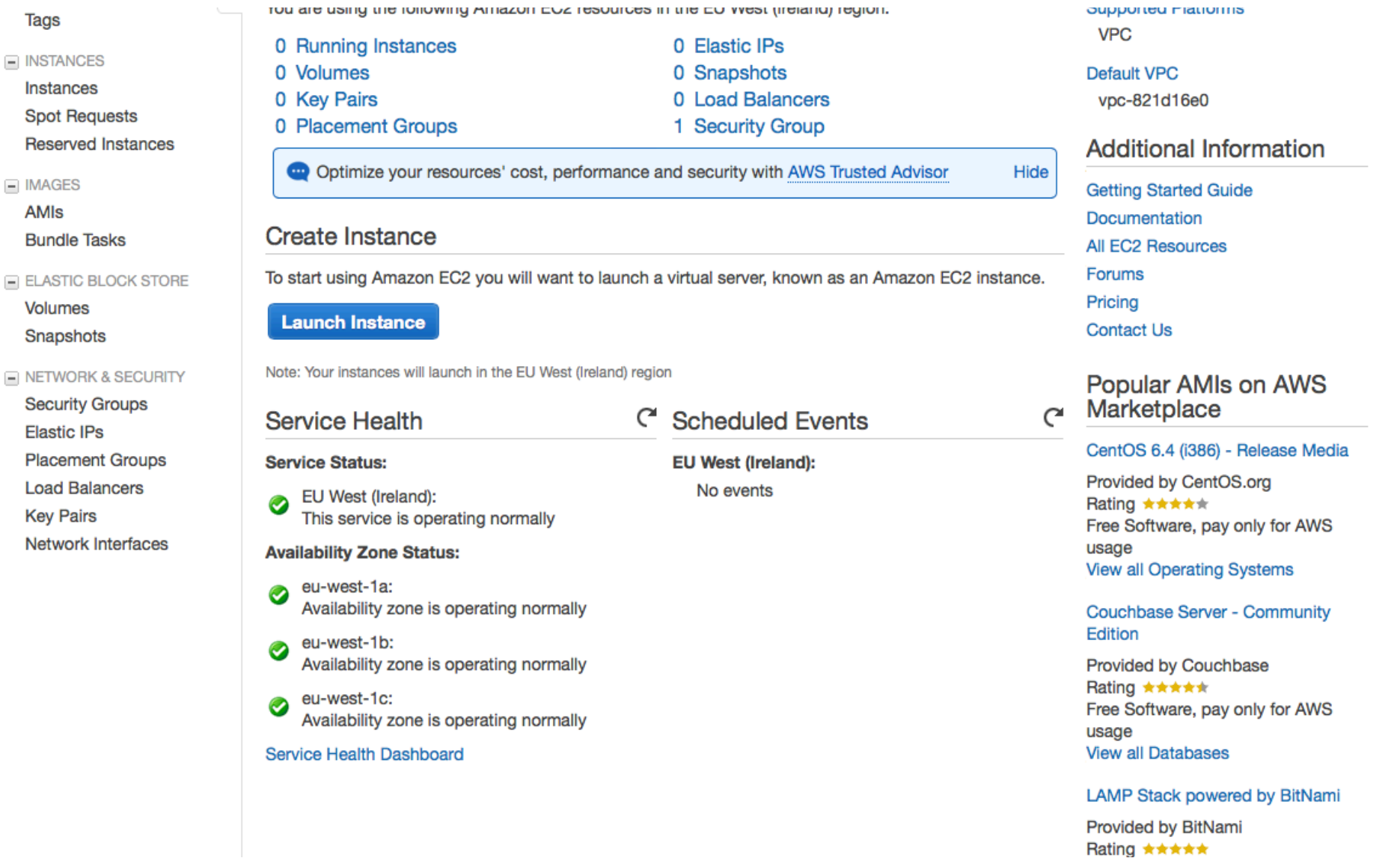
Sign-in EC2 for a Free Account

- Caveats
  - You need a “valid” visa number
  - You need a “valid” phone number to verify your identity
- Provides
  - 750 hours of micro Linux/Windows instances per month
  - 30GB in EBS

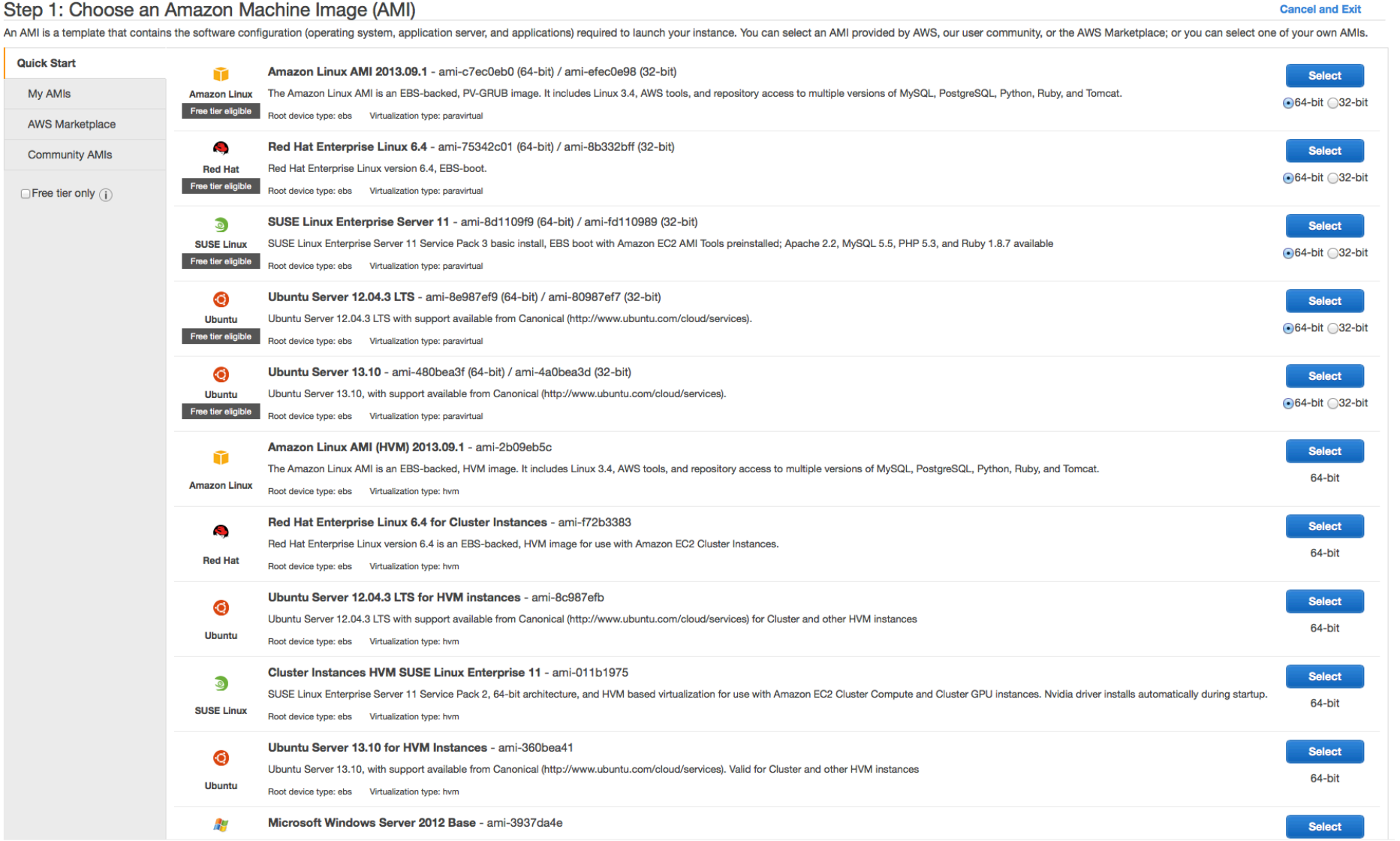
AWS management console



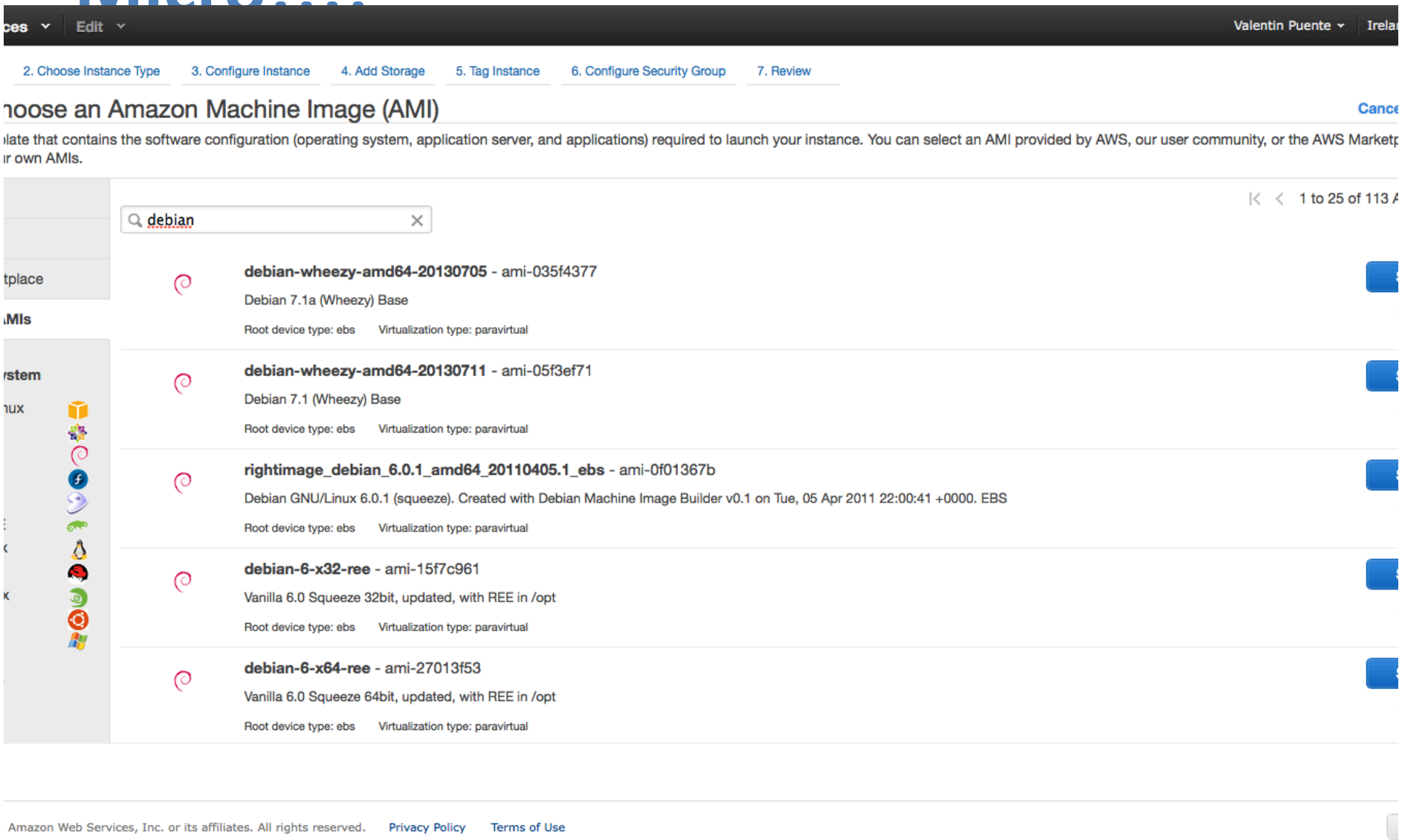
EC2 management Console



Create a new server (using preset-AMI)



Community AMIs (Debian) Use Micro!!!!

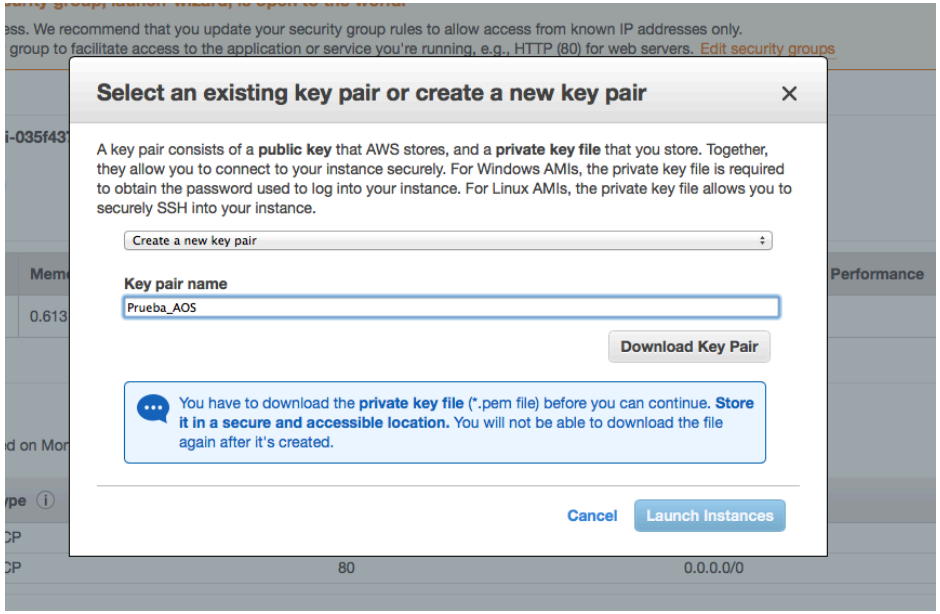


Configure instance

- 2 Instance type
  - Use a micro instance (otherwise you will pay)
- 3 Configure instance
  - Spot Instances
    - Like stock options buy: limit your maximum price
  - Assign a public IP to the instance
  - Multi-tenancy
    - Book a “private” hw or not (and pay more)
  - Kernels and ramdisk boot
    - Only change if you know what you need: LVM, iSCSI, etc…?
- 4 Storage
  - 8GB is enough
- 6 Security options
  - Firewall
    - How its accessible the machine and from where?
  - SSH config avaliable!!!
    - Otherwise, you will be not able to access to it!

Security Keys

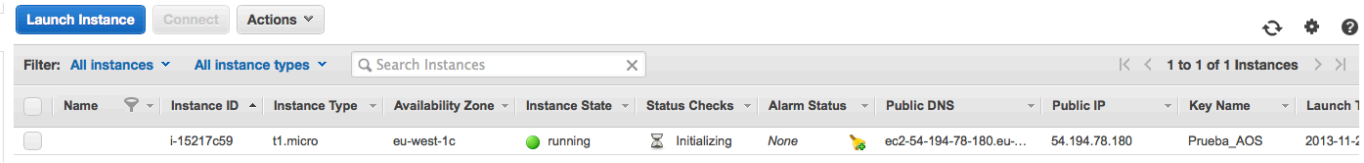
- Ssh access is done by key pair



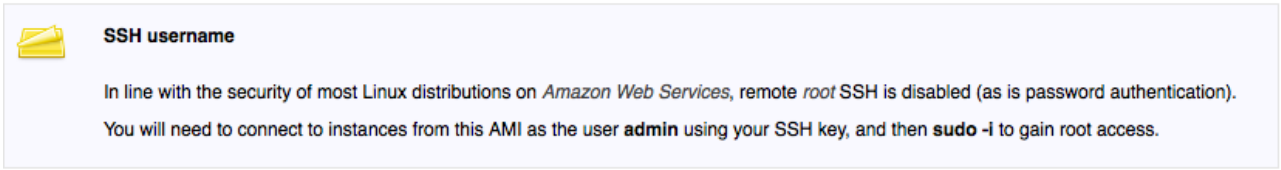
- Download the key and keep it safe!
  - # mv Downloads/prueba_AOS.pem $HOME/.ssh/aws.pem
  - # chmod go-rwx $HOME/.ssh/aws.pem

Accessing your instance

- Check in console when is up and running

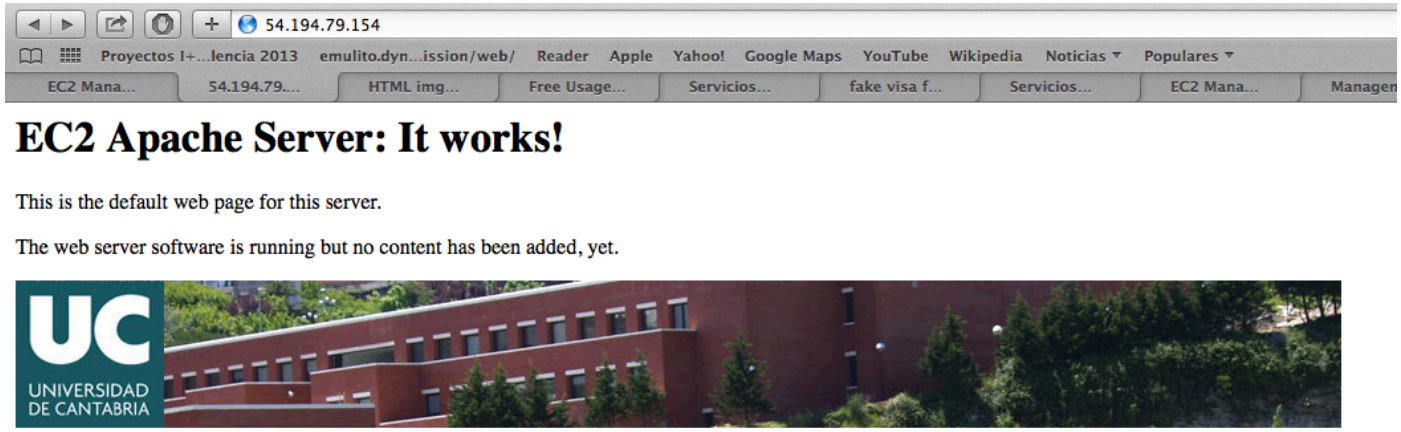


- Log-in your brand new machine and install apache
  - # ssh –i $HOME/.ssh/aws.pem **admin**@IP_public
- Never connect as root
  - In this case we are using a “official” EC2 Image/AMI
    - https://wiki.debian.org/Cloud/AmazonEC2Image/Wheezy



Install a new Apache2 Server

- As usual
  - #sudo su –
  - #apt-get update
  - #apt-get install apache2
- We have a “real” web server listening in the public_IP



Amazon Machine Images

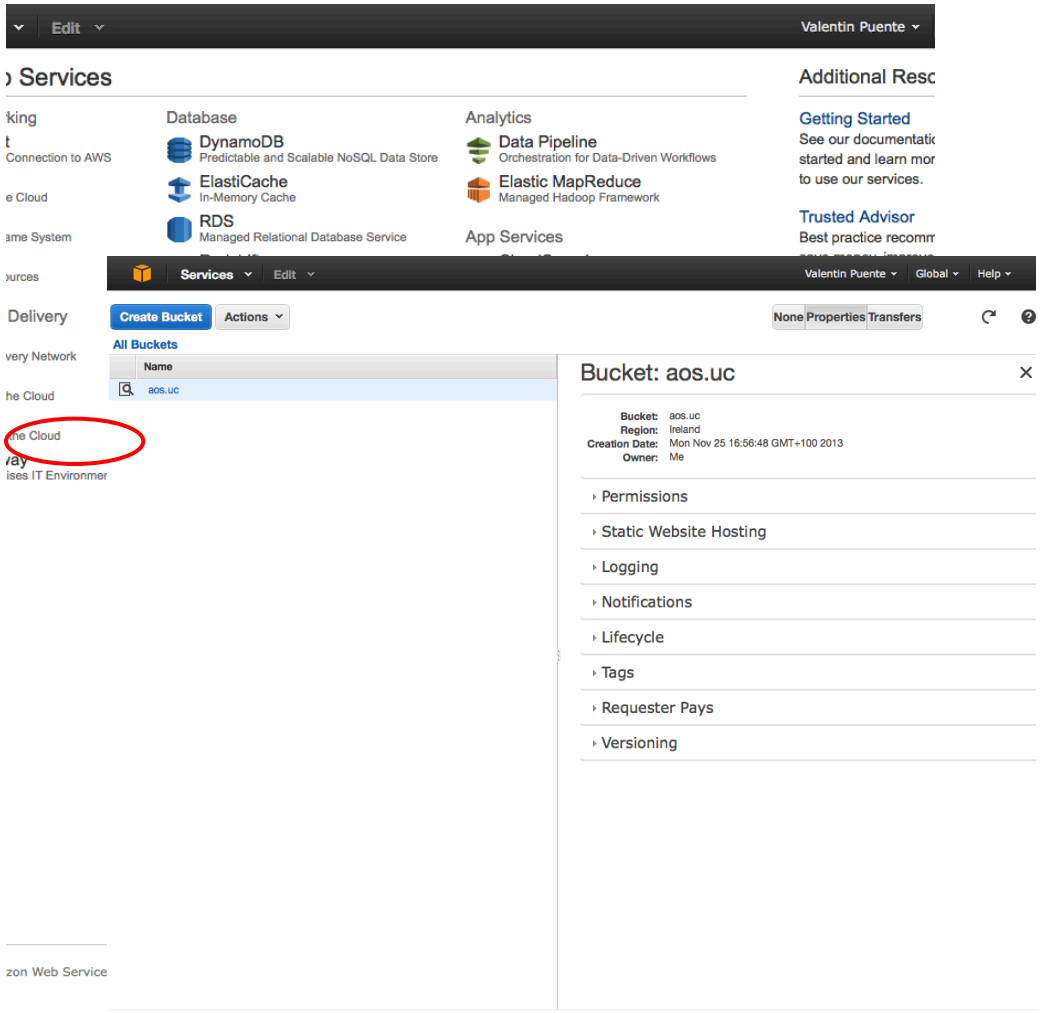
- It's a boot disk
  - Pre-bundled OS+Tools installed
  - Should not contain "data"
- You can use
  - AMIs from Amazon
  - AMIs with 3rd parties software
  - AMIs from the community…
  - Your own AMIs
- Public AMIs
  - **Ups**: Support, Community, maintenance, etc…
  - **Downs**: Security, Documentation, Software setup, etc..

Previous: concepts

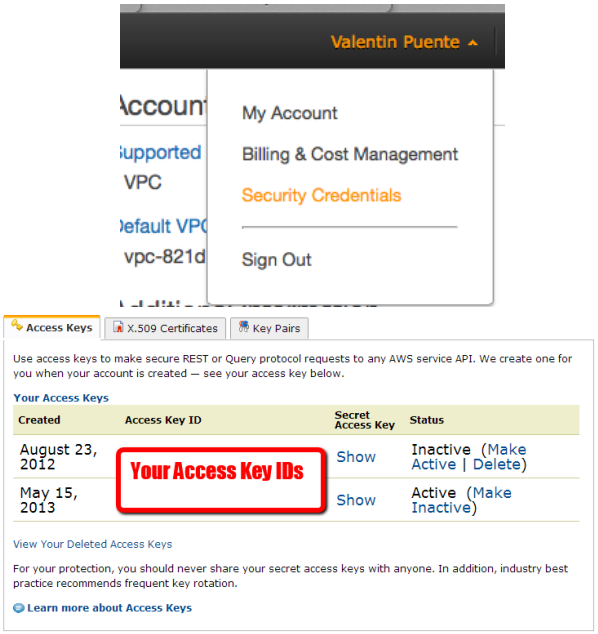
- Bucket
  - possibly public, globally unique name; contains multiple images
- Prefix
  - possibly public, globally unique name; contains multiple images
- Image ID (AMI ID)
  - **Unique** identifier for an individual image; assigned by EC2
  - Required to create a new instance

Previous

Create a Bucket in S3



Get your credentials (IAM) AWS_ACCESS_KEY and AWS_ACCESS_KEY



Task1: Custom AMI from a pre-bundled Instance

- Steps
  - 1 Configure a prebuilt instance
    - Installing additional software, configurations, etc…
  - 2 Rebundle and upload to S3
  - 3 Register as a new AMI
- Tools required
  - EC2 AMI tools: http://aws.amazon.com/developertools/368
    - Use alien to convert rpm into deb
  - AWS CLI python frontend
    - Installed with pip

Rebundle a running AMI and upload it to S3

#Create a X.509 certificate/private key in Security Credentials (IAM)

#Upload both to the EC2 instance

scp -i $HOME/aws.pem Downloads/*.pem admin@<EC2_pubIP>:/tmp/

#Connect

ssh –i $HOME/aws.pem admin@<EC2_pubIP>

#Install EC2 AMI tools and install other tools

wget rpm from http://aws.amazon.com/developertools/368

apt-get install alien ruby curl && alien ec2-ami-tools.noarch.rpm && alien ec2-ami-tools.noarch.rpm

#Prepare Ruby

export RUBYLIB=$RUBYLIB:/usr/lib/ruby/site_ruby

#Bundle the image (You can get user ID from IAM in "Account identification"

ec2-bundle-vol --prefix myDebian --user xxxx-xxxx-xxxx --destination /mnt --privatekey /tmp/pk-*.pem --cert /tmp/cert-*.pem --exclude /mnt/,/tmp --arch x86_64

#Upload the bundle to S3 (Need AWS_ACCESS_KEY and AWS_ACCESS_KEY from IM)

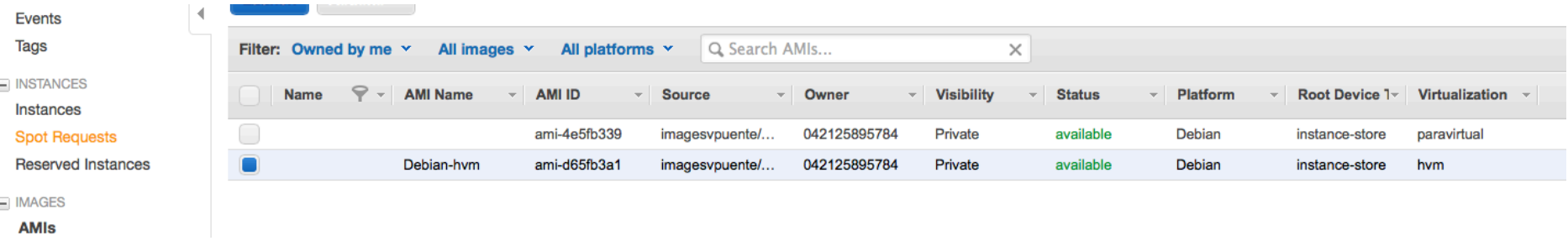
Beware bucketnames (aos.uc gives a SSL error uses something without ".")

ec2-upload-bundle --bucket imagesVpuente -m /mnt/myDebian.manifest.xml --access-key XXXXXX --secret-key xxxxxxxxx

Register the new AMI

- More powerful than old EC2 CLI
  - Install aws cli tools (http://aws.amazon.com/es/cli/)
    - apt-get install -y python-pip
    - pip install awscli
  - Edit $HOME/.aws-config or “aws configure”

    [default]
    aws_access_key_id = XXXXXXXX
    aws_secret_access_key = xxxxxxxx region = eu-west-1
    export AWS_CONFIG_FILE=$HOME/.aws-config
- Register the AMI (you can change a lot of properties, such as VM type)
  - aws ec2 register-image --image-location imagesvpuente/myDebian.manifest.xml
  - aws ec2 register-image --image-location imagesvpuente/myDebian.manifest.xml --virtualization-type hvm --name Debian-hvm
- Now Available in my personal AMIs (free tier is not available!)



Uploading your own VM I (Windows on ~~Xen~~)

- EC2 Supports Import/Export from Vmware, Hyper-V, Xen, etc… through EC2-api-tools
- Recipe
  - 1 Install API tools
    - wget http://launchpadlibrarian.net/111617788/ec2-api-tools_1.6.1.1-0ubuntu1_all.deb
    - dpkg -i ec2-api-tools_1.6.1.1-0ubuntu1_all.deb
  - Get your AWS_ACCESS_KEY and AWS_ACCESS_KEY from top EC2 console, User Name →Security Credentials)
  - #export AWS_ACCESS_KEY=XXXXXXXXXXXXXXXXXXXXX
  - #export AWS_SECRET_KEY=XXXXXXXXXXXXXXXXXXXXX

Uploading your own VM 2

- Create EC2 using RAW Xen disk in a prebuilt S3 bucket (aos.uc) (Note: Sync clocks with ntpdate)
  - ec2-import-instance /vms/images/image.img -f RAW -t t1.micro -a x86_64 -b aos.uc -o $AWS_ACCESS_KEY -w $AWS_SECRET_KEY --region eu-west-1
- Check the progress
  - ec2-describe-conversion-tasks import-i-ffm2nty0 -O $AWS_ACCESS_KEY -W $AWS_SECRET_KEY --region eu-west-1
- Only valid for Windows Images (No Linux support at this time)