



AMAZON WEB SERVICES WITH GRID GPUS

January 2015



Getting Started Guide to using AWS
Linux AMIs with NVIDIA GRID



DOCUMENT CHANGE HISTORY

Version	Date	Authors	Description of Change
0.1	2/7/2014	EY	Initial draft (separate the Windows and Linux guides)
0.2	2/13/2014	EY	Updated guide with Ubuntu 12.04
0.5	11/14/2014	EY	Guide revised adding Ubuntu 14.04 and removing 12.04

CONTENTS

Setting up an Amazon AWS account	4
Launching G2 instances with Amazon EC2 web-based interface console	5
Step 1: Create a key pair to connect remotely with TeamViewer/SSH	5
Step 2: Launching AMIs from the EC2 Management Console	6
Step 3: Launching Linux AMIs from the EC2 Management Console	7
Additional Network Configuration of the	8
Step 6 - Connect to a Linux Instance using SSH or PuTTY	9
Option 1 - Setting up PuTTY Secure Copy Client	11
Step 8 - Configure AMI and install the NVIDIA Graphics Driver	15
 Ubuntu Server 14.04 LTS (HVM) - ami-3d50120d	15
 Amazon Linux Amazon Linux AMI (CentOS) installation	17
Step 9: Remotely Connect to the Linux AMI (NVIDIA driver installed)	18
Configuring NVIDIA Linux Driver	18
Configuring Linux AMI (Ubuntu 14.04 LTS with HVM)	21
Optional: Installing the Amazon EC2 Command Line Interface Tools	22
Step 1: Download the Command Line Interface Tools (CLI Tools)	22
Step 2: Setup the JAVA_HOME Environment Variable	23
Step 3: Setup the EC2_HOME Environment Variable	25
Step 4: Setup the EC2_URL Environment Variable	26
Step 5: Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables	26
Step 6: Set the Region (Optional)	28
Step 7: Use a Proxy (Optional)	29
Working with instances	30
Create a key-pair	30
Creating a Security Group	32
Add rules for security group	32
Create Instances	33
Starting Instances	34
Stopping Instances	34
Rebooting Instances	34
Terminating Instances	34
Connecting to Instances	35

Setting up an Amazon AWS account

If you have an AWS account, you can skip this part and go to the next page.

To register an AWS account, you need a valid email address and a credit card to pay for the use of AWS servers by the hour. Click on this [EC2 Console](#) to go to the webpage to register for a new AWS account. Here you will enter your E-mail address to register, and select “**I am a new user**”. Now click on the button “Sign in using our secure server” to begin account registration.



Sign In or Create an AWS Account

You may sign in using your existing Amazon.com account or you can create a new account by selecting “I am a new user.”

My e-mail address is:

☒ I am a new user.

☐ I am a returning user
and my password is:

[Sign in using our secure server](#)

[Forgot your password?](#)

[Has your e-mail address changed?](#)

Learn more about [AWS Identity and Access Management](#) and [AWS Multi-Factor Authentication](#), features that provide additional security for your AWS Account.

You will be asked to fill in the registration form for additional information. You must provide password, name, address, company, and a credit card for payment. Afterwards, you will receive a confirmation e-mail once everything has been confirmed.

There are two ways to manage Amazon EC2 instances, one through the web interface, and another through the Command Line API tools. This document will cover both methods.

Launching G2 instances with Amazon EC2 web-based interface console

AWS EC2 Management Console is a point-and-click web-based interface, you can easily use your browser to launch, start or stop your instance. Log in [EC2 Console](#) with your registered E-mail and password, completing the following tasks to set up a G2 instance:

1. Create a key pair
2. Launch an AMI from AWS EC2 console
3. Connect to the instance by TeamViewer or SSH
4. Setup VNC connection with NVIDIA Graphics Driver

Step 1: Create a key pair to connect remotely with TeamViewer/SSH

First create a key pair so that you can connect to Amazon EC2 instances. Public AMI instances use a public/private key pair to log in rather than a password. This public key is embedded in your instance, allowing you who has a private key to log in securely without a password. After creating your own AMIs, you can choose other mechanisms to securely log into your new instances. Next we will use AWS Management Console to create a key pair.

To generate a key pair

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the top navigation bar, in the region selector and choose the location that is closest to you. Note there are different prices for different locations. In this case we will choose **US West (Oregon)**.
3. In the left navigation pane, under Network and Security, click Key Pairs.
4. Click Create Key Pair.
5. Type **mykeypair** in the new Key pair name box, and then click **Yes**.
6. Download the private key file, which is named **mykeypair.pem**, and keep it in a safe place. You will need it to access any instances that you launch with this key pair. Keep this key pair in a safe place, because if it is lost, you will not be able to connect to your Amazon EC2 instances

Note: For more information about key pairs, see [Amazon EC2 Key Pairs](#) in the Amazon Elastic Compute Cloud User Guide.

Since network traffic to the server will change based on demand, AWS must be configured to properly scale the number instances when appropriate. In order to do this, you want this server to use the [Auto Scaling](#) feature to create an Auto Scaling group. It is necessary to create the Auto Scaling group and associate our Auto Scaling group with our Elastic Load Balancer

Step 2: Launching AMIs from the EC2 Management Console

Before launching your instance, you must choose the appropriate EC2 location. On the upper right, moving over the menu allows you choose the region you want the instance to be launched from. In this case, we use **US West (Oregon)**.

The screenshot displays the AWS Management Console for the EC2 service. On the left is a navigation sidebar with categories like EC2 Dashboard, INSTANCES, IMAGES, ELASTIC BLOCK STORE, NETWORK & SECURITY, and AUTO SCALING. The main content area is titled 'Resources' and shows a summary of EC2 resources in the 'US West (Oregon)' region: 0 Running Instances, 1 Volume, 1 Key Pair, 0 Placement Groups, 0 Elastic IPs, 0 Snapshots, 0 Load Balancers, and 3 Security Groups. Below this is a 'Create Instance' section with a 'Launch Instance' button and a note that instances will launch in the US West (Oregon) region. To the right of the 'Create Instance' section is a 'Service Health' table showing the status of the US West (Oregon) region and its availability zones (us-west-2a, us-west-2b, us-west-2c), all of which are operating normally. Further right is a 'Scheduled Events' section showing no events for the US West (Oregon) region. On the far right, there is a dropdown menu for selecting the region, with 'US West (Oregon)' selected, and a list of popular AMIs on the Marketplace.

Resources

You are using the following Amazon EC2 resources in the US West (Oregon) region:

- 0 Running Instances
- 1 Volume
- 1 Key Pair
- 0 Placement Groups
- 0 Elastic IPs
- 0 Snapshots
- 0 Load Balancers
- 3 Security Groups

Optimize your resources' cost, performance and security with [AWS Trusted Advisor](#) [Hide](#)

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

[Launch Instance](#)

Note: Your instances will launch in the US West (Oregon) region

Service Health

Service Status:

- US West (Oregon): This service is operating normally

Availability Zone Status:

- us-west-2a: Availability zone is operating normally
- us-west-2b: Availability zone is operating normally
- us-west-2c: Availability zone is operating normally

Scheduled Events

US West (Oregon): No events

Popular AMIs on AWS Marketplace

- Vyatta Virtual Router/Firewall/VPN
- Alert Logic Threat Manager for AWS
- Ardohe ColdFusion

Click on the Launch Instance button from the EC2 Management Console to get started. *Note: You can change which Data Center location you would like your instance to launch from. You can also configure the instance to use multiple regions provided that G2 instances are available in these other regions. Please click on [Regions and Availability Zones](#) for more information.*

Create Instance





To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

[Launch Instance](#)




Note: Your instances will launch in the US West (Oregon) region

Step 3: Launching Linux AMIs from the EC2 Management Console

- a) Ubuntu Server AMI (Ubuntu 14.04 LTS for HVM Instances), just scroll down the list and choose AMI ID (ami-3d50120d).

 Amazon Linux Free tier eligible	Amazon Linux AMI 2014.09.1 (HVM) - ami-b5a7ea85 The Amazon Linux AMI is an EBS backed image. It includes the 3.14 kernel, Ruby 2.1, PHP 5.5, PostgreSQL 9.3, Docker 1.2, the AWS command line tools, and repository access to many other packages. Root device type: ebs Virtualization type: hvm	Select 64-bit
 Red Hat Free tier eligible	Red Hat Enterprise Linux 7.0 (HVM), SSD Volume Type - ami-99bef1a9 Red Hat Enterprise Linux version 7.0 (HVM), EBS General Purpose (SSD) Volume Type Root device type: ebs Virtualization type: hvm	Select 64-bit
 SUSE Linux Free tier eligible	SuSE Linux Enterprise Server 12 (HVM), SSD Volume Type - ami-d7450be7 SuSE Linux Enterprise Server 12 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled. Root device type: ebs Virtualization type: hvm	Select 64-bit
 Ubuntu Free tier eligible	Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-3d50120d Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services). Root device type: ebs Virtualization type: hvm	Select 64-bit

- b) Amazon Linux AMI instance is based upon CentOS. In the search box type “NVIDIA GRID”, select the Amazon Linux AMI with NVIDIA GRID GPU Driver.

 NVIDIA.	Windows Server 2008R2 with NVIDIA GRID GPU Driver ★★★★★ (1) 1.0 Sold by NVIDIA \$0.00/hr for software + Charges for EC2 with Windows + AWS usage fees Windows, Windows 2008 R2 6.1 64-bit Amazon Machine Image (AMI) Updated: 10/15/13 Amazon EC2 running Microsoft Windows Server is a fast and dependable environment for deploying applications using the Microsoft Web Platform. The NVIDIA GRID GPU Driver ... More info	Select
 NVIDIA.	Windows Server 2012 with NVIDIA GRID GPU Driver ★★★★★ (7) 1.0 Sold by NVIDIA \$0.00/hr for software + Charges for EC2 with Windows + AWS usage fees Windows, Windows Server 2012 6.2 64-bit Amazon Machine Image (AMI) Updated: 10/15/13 Amazon EC2 running Microsoft Windows Server is a fast and dependable environment for deploying applications using the Microsoft Web Platform. The NVIDIA GRID GPU Driver ... More info	Select
 NVIDIA.	Amazon Linux AMI with NVIDIA GRID GPU Driver ★★★★★ (4) 2014.09.1 Previous versions Sold by NVIDIA Corporation \$0.00/hr for software + AWS usage fees Linux/Unix, Amazon Linux 2014.09 64-bit Amazon Machine Image (AMI) Updated: 10/15/14 The Amazon Linux AMI is a supported and maintained Linux image provided by Amazon Web Services for use on Amazon Elastic Compute Cloud (Amazon EC2). It provides a stable, ... More info	Select

Step 4: Launching a Linux AMI

Now click on the Select button. The web page will be updated to allow you to choose the instance type (GPU instances), which has the **g2.2xlarge** instance type. Click on **g2.2xlarge**, and under **Currently selected** of the console, **g2.2xlarge** has been selected. These Windows or Linux Server instances offer the NVIDIA GRID (Kepler GK104) GPUs. Click on the **Review and Launch** button to start the session.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Currently selected: g2.2xlarge (26 ECUs, 8 vCPUs, 15 GiB memory, 1 x 60 GiB Storage Capacity)

All instance types	GPU instances
Micro instances Free tier eligible	GPU instances provide graphics processing units (GPUs) along with high CPU and network performance for applications benefiting from highly parallelized processing, including 3D graphics, HPC, rendering, and media processing applications.
General purpose	
Memory optimized	
Storage optimized	
Compute optimized	
GPU instances	

Size	ECUs ⓘ	vCPUs ⓘ	Memory (GiB)	Instance Storage (GiB) ⓘ	EBS-Optimized Available ⓘ	Network Performance ⓘ
g2.2xlarge	26	8	15	1 x 60 (SSD)	Yes	High

G2 Instances are backed by 1 x NVIDIA GRID GPU (Kepler GK104) and 8 x hardware hyperthreads from an Intel Xeon E5-2670

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#) [Feedback](#)

You will confirm this is the correct AMI, and then click on the button **Review and Launch**. It will prompt you for the *keypair*, where you will choose the previously created one. Provide the local path for where the file that has key your machine. Then check the box acknowledging that you have access to the selected private key file. Click on the **Launch Instances** to launch your Amazon G2 Server instance.

Additional Network Configuration of the

Linux instances will need a SSH connection during setup. Make sure a SSH port has been opened in your current security group. Click **Security Groups** in the left navigation pane under **NETWORK & SECURITY**. Choose the security group bound to your current instance, on the **Inbound** tab, select **SSH** from **Create a new rule**, leave **Source** as 0.0.0.0/0, and then click **Add Rule** and **Apply Rule Changes** to add the rule.

Details **Inbound***

Create a new rule: SSH

Source: 0.0.0.0/0
(e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

[Add Rule](#)

Your changes have not been applied yet.

[Apply Rule Changes](#)

TCP Port (Service)	Source	Action
80 (HTTP)	0.0.0.0/0	Delete
22 (SSH)	0.0.0.0/0	Delete

Step 6 – Connect to a Linux Instance using SSH or PuTTY

Connecting via SSH on Linux

If you are connecting to a Linux instance from a client computer running Linux, you will need to specify the **mykeypair.pem** file to your SSH client. Username is usually **ec2-user** on EC2 instances. Also make sure your file **mykeypair.pem** only read/write access by the original user. To do this run `'chmod 600 mykeypair.pem'`

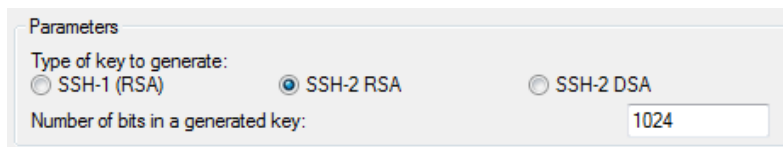
```
ssh -l username -i mykeypair.pem <public IP Address of Linux Instance>
```

Connecting via PuTTY for Windows

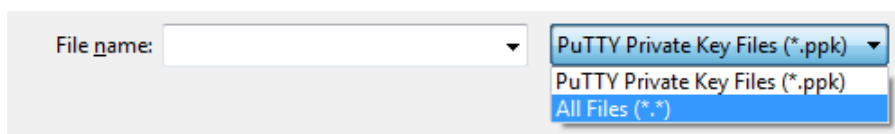
If connecting to a Linux instance from a local Windows computer, you can use either MindTerm or PuTTY. If you plan to use PuTTY, you'll need to install it and use the following procedure to convert the `.pem` file to a `.ppk` file.

Download and install PuTTY (for SSH) from <http://www.chiark.greenend.org.uk/~sgtatham/putty/>. Make sure that the entire PuTTY suite is installed

1. Start PuTTYgen (for example, from the Start menu, select All Programs > PuTTY > PuTTYgen).
2. Under the Parameter for the Type of key to generate, choose SSH-2 RSA.



3. Click Load. By default, PuTTYgen displays only files with the extension `.ppk`. To locate your `.pem` file, select the option to display files of all types.



4. Select the private key file that you created in the previous procedure and click Open. Click OK to dismiss the confirmation dialog box.
5. Now click on Save private key. PuTTYgen will display a warning about saving the key without a passphrase. Click on Yes.

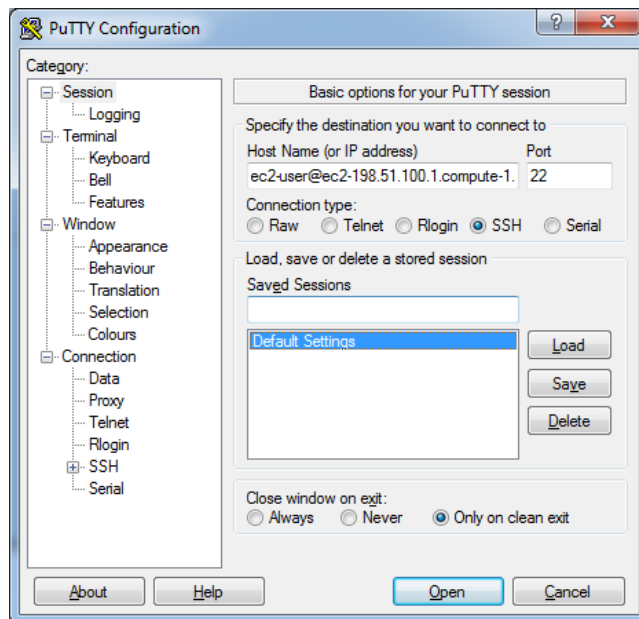
6. Specify the same name for the key used for the key pair (i.e. for example, *your_user_name-key-pair-region_name*). PuTTY automatically adds the .ppk file extension.

Connect to a Linux instance using PuTTY for Windows

PuTTY doesn't use .pem files, it uses .ppk files. If you haven't already generated a .ppk file, do so now using PuTTYgen. For more information, see [To prepare to connect to a Linux instance from Windows using PuTTY](#).

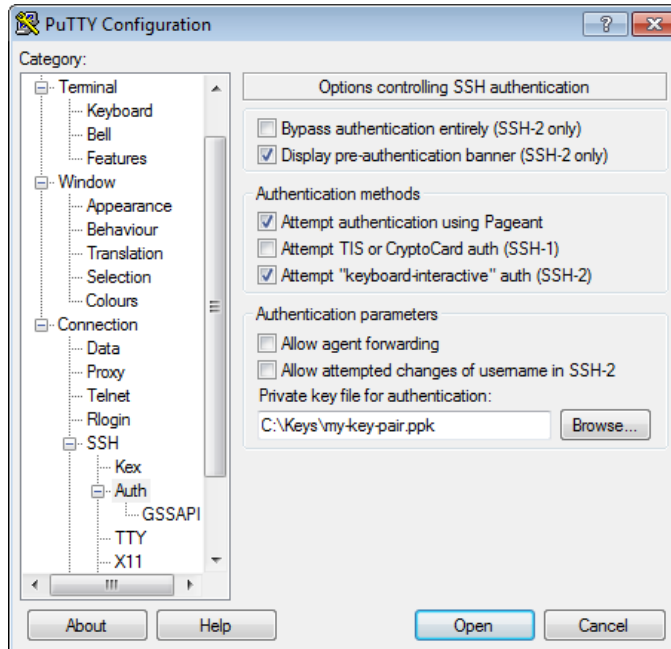
Public DNS can be found from **Description** tab in **Security Groups** by clicking on **Instances** in the left navigation pane under **INSTANCES**.

1. Start PuTTY (from the Start menu, click All Programs > PuTTY > PuTTY).
2. In the Category pane, select Session and complete the following fields:
 - a. In the Host Name box, enter `ec2-user@public_dns_name`.
 - b. Under Connection type, select SSH.
 - c. Ensure that Port is 22.



- d. In the Category pane, expand Connection, expand SSH, and then select Auth. Complete the following:
 1. Click Browse, Select the .ppk file that you generated from your key pair.

2. Click Open to start the PuTTY session.



3. If this is the first time you have connected to this instance, PuTTY will display a security alert dialog box that asks whether you trust the host you are connecting to. Click Yes. A window opens and you are connected to your instance.

Option 1 - Setting up PuTTY Secure Client

The PuTTY Secure Copy client (PSCP) is a command-line tool that you can use to transfer files between your Windows computer and your Linux/UNIX instance. If you prefer a graphical user interface (GUI), you can use an open source GUI tool named WinSCP. For more information, see [Transferring Files to Your Instance with WinSCP](#).

To use PSCP, you'll need the private key you generated in [Converting Your Private Key Using PuTTYgen](#). You'll also need the public DNS address of your Linux/UNIX instance.

The following example transfers the file `Sample_file.txt` from a Windows computer to the `/usr/local` directory on a Linux/UNIX instance:

```
C:\> pscp -i C:\Keys\my-key-pair.ppk C:\Sample_file.txt
user_name@public_dns:/usr/local/Sample_file.txt
```

Option 2: Transferring Files using WinSCP

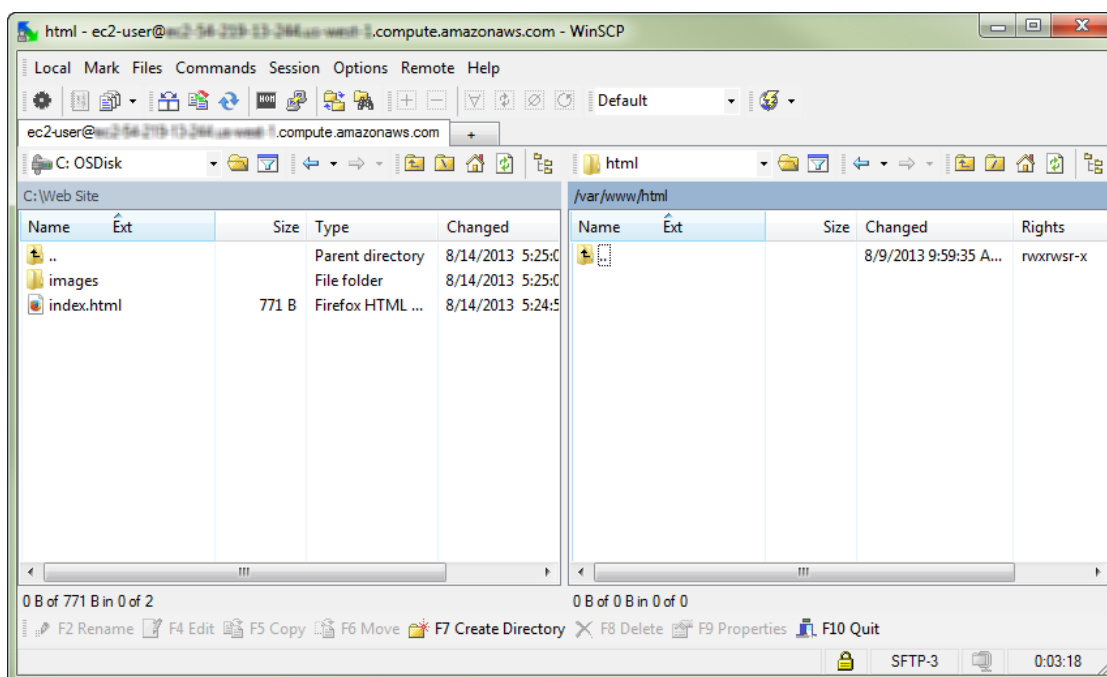
WinSCP is a GUI-based file manager for Windows that allows you to upload and transfer files to a remote computer using the SFTP, SCP, FTP, and FTPS protocols. WinSCP allows you to drag and drop files from your Windows machine to your Linux instance or synchronize entire directory structures between the two systems.

To use WinSCP, you will need the private key you generated in [Converting Your Private Key Using PuTTYgen](#). You will also need the public DNS address of your Linux/UNIX instance.

1. Download and install WinSCP from <http://winscp.net/eng/download.php>. For most users the default installation options are OK.
2. Start WinSCP.
3. At the WinSCP login screen, for Host name, enter the public DNS address of your instance.
4. Under User name, enter the default user name for your AMI. For Amazon Linux AMIs, the user name is **ec2-user**. For Red Hat AMIs the user name is **root**, and for Ubuntu AMIs the user name is **ubuntu**.
5. Enter the path of this private key. Click the "..." button to browse for the file.

Note: WinSCP requires a PuTTY private key file (.ppk). You can convert a .pem security key file to the .ppk format using PuTTYgen. For more information, see [Converting Your Private Key Using PuTTYgen](#).

6. (Optional) In the left panel, click Directories, and then, for Remote directory, enter the path for the directory you want to add files to.
7. Click Login to connect, and click Yes to add the host fingerprint to the host cache.



8. After the connection is established, you can find the Linux instance in the connection window on the right. Your local machine is going to be on the left. You can drag and drop files directly into the remote file system from your local machine. For more information on WinSCP, see the project documentation at <http://winscp.net/eng/docs/start>.

Option 3: Connecting using a Linux SSH Client from a local machine

Your Linux computer most likely includes an SSH client by default. You can check for an SSH client by typing **ssh** at the command line. If your computer doesn't recognize the command, the OpenSSH project provides a free implementation of the full suite of SSH tools. For more information, see <http://www.openssh.org>.

Open your command shell and run the following command:

```
ssh -i /path/key_pair.pem ec2-user@public_dns_name
```

Tip: For Amazon Linux, the user name is **ec2-user**. For RHEL5, the user name is often **root** but might be **ec2-user**. For an Ubuntu, AMI the user name is **ubuntu**. Otherwise, check with your AMI provider.

Transferring Files Linux/UNIX Instances from Linux/UNIX with SCP

One way to transfer files between your local computer and a Linux/UNIX instance is to use Secure Copy (SCP). This section describes how to transfer files with SCP. The procedure is very similar to the procedure for connecting to an instance with SSH. Most Linux, UNIX, and Apple computers include an SCP client by default. If yours doesn't, the OpenSSH project provides a free implementation of the full suite of SSH tools, including an SCP client. For more information, go to <http://www.openssh.org>.

Transfer a file to your instance using the instance's public DNS name. For example, if the name of the private key file is `my-key-pair`, the file to transfer is `SampleFile.txt`, and the public DNS name of the instance is `ec2-198-51-100-1.compute-1.amazonaws.com`, use the following command to copy the file to the `ec2-user` home directory.

`scp -i my-key-pair.pem SampleFile.txt ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com:~`You'll see a response like the following.

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com
(10.254.142.33)'
can't be established.
RSA key fingerprint is
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

Select “**yes**” , You will see a response like the following.

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)
to the list of known hosts.
Sending file modes: C0644 20 SampleFile.txt
Sink: C0644 20 SampleFile.txt
SampleFile.txt          100%   20    0.0KB/s   00:00
```

To transfer files in the other direction (from your Amazon EC2 instance to your local computer), simply reverse the order of the host parameters. For example, to transfer the SampleFile.txt file from your Amazon EC2 instance back to the home directory on your local computer as SampleFile2.txt, use the following command on your local computer.

```
scp -i my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-
1.amazonaws.com:~/SampleFile.txt ~/SampleFile2.txt
```

Step 8 – Configuring the Linux AMI and the NVIDIA Graphics Driver



Ubuntu Server 14.04 LTS (HVM) - ami-3d50120d

First install the following Linux libraries and dependencies:

```
ubuntu@ip-10-236-190-141:~$ sudo apt-get update
ubuntu@ip-10-236-190-141:~$ sudo apt-get install linux-image-generic linux-
image-extra-virtual linux-image-extra-$(uname -r) linux-headers-$(uname -r)
linux-source*
ubuntu@ip-10-236-190-141:~$ sudo apt-get install build-essential gcc g++ gcc-
multilib lib32z1 lib32ncurses5 lib32bz2-1.0 lib32stdc++6
ubuntu@ip-10-236-190-141:~$ sudo apt-get install unzip make pkg-config mesa-
utils libgl1-mesa-glx libglu1-mesa libglu1-mesa-dev libglew-dev freeglut3-dev
libx11-dev libxmu-dev libxi-dev libXxf86vm-dev ubuntu-desktop x11vnc xserver-
xorg
```

Remove the nouveau driver and blacklist it using the instructions below:

```
ubuntu@ip-10-236-190-141:~$ sudo apt-get purge nvidia*
ubuntu@ip-10-236-190-141:~$ sudo apt-get purge xserver-xorg-video-nouveau
ubuntu@ip-10-236-190-141:~$ sudo vim /etc/modprobe.d/blacklist.conf

# Add to the end of this file and save the blacklist.conf file ":wq!"
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv

# update the boot initrd, and reboot
ubuntu@ip-10-236-190-141:~$ sudo update-initramfs -u
ubuntu@ip-10-236-190-141:~$ sudo reboot
```

Create the symbolic links for GLEW, X11, and other needed libraries:

```
# First create symbolic links for i386
ubuntu@ip-10-236-190-141:~$ cd /usr/lib/i386-linux-gnu
ubuntu@ip-10-236-190-141:~$ sudo ln -s libGLU.so.1 libGLU.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libGLEW.so.1.6 libGLEW.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libGLEWmx.so.1.6 libGLEWmx.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libXi.so.6 libXi.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libX11.so.6 libX11.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libXrandr.so.2 libXrandr.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libXxf86vm.so.1 libXxf86vm.so
ubuntu@ip-10-236-190-141:~$ sudo ln -s libstdc++.so.6 libstdc++.so

# Now repeat creating symbolic links for x86_64
ubuntu@ip-10-236-190-141:~$ cd /usr/lib/x86_64-linux-gnu
```

To install the NVIDIA GRID driver, you must be **root** or **sudo** to do so. The driver can be downloaded from [here](#). Install the NVIDIA driver. When prompted, choose to install the 32-bit OpenGL compatibility libraries.

```
ubuntu@ip-10-236-190-141:~/GRID$ cd drivers
ubuntu@ip-10-236-190-141: ~/GRID$ sudo service lightdm stop
ubuntu@ip-10-236-190-141: ~/GRID$ sudo pkill X
ubuntu@ip-10-236-190-141: ~/GRID$ chmod +x NVIDIA*
ubuntu@ip-10-236-190-141:~/GRID/drivers$ sudo sh NVIDIA-Linux-x86_64-
xxx.xx.run
```




Amazon Linux AMI (CentOS) installation

Install the following Linux libraries and dependencies prior to installing the NVIDIA driver.

```
ec2-user@ip-10-236-190-141:~$ sudo yum groupinstall "Development Tools"
ec2-user@ip-10-236-190-141:~$ sudo yum install kernel-devel kernel-headers dkms
```

Now blacklist nouveau by adding `"blacklist nouveau"` to this file without the quotes.

```
ec2-user@ip-10-236-190-141:~$ sudo vim /etc/modprobe.d/blacklist.conf
```

Create a new `"initramfs"` file and taking backup of the existing one

```
ec2-user@ip-10-236-190-141:~$ sudo mv /boot/initramfs-$(uname -r).img
/boot/initramfs-$(uname -r).img.bak
ec2-user@ip-10-236-190-141:~$ sudo dracut -v /boot/initramfs-$(uname -r).img
$(uname -r)
ec2-user@ip-10-236-190-141:~$ sudo reboot
```

To install the NVIDIA GRID driver, you must be **root** or **sudo** to do so. The driver can be found [here](#). Here we assume that the user created a folder `"grid"` under the user directory. In this case, we will just assume the `NVIDIACaptureSDK_Installation` is `"grid"`. Now install the NVIDIA driver. Make sure to install the 32-bit compatibility libraries for OpenGL when prompted.

```
ec2-user@ip-10-236-190-141:~$ sudo sh NVIDIA-Linux-x86_64-xxx.xx.run
```

Now let's create the symbolic links for GLEW and other required libraries.

```
# First create symbolic links for i386
ec2-user@ip-10-236-190-141:~$ cd /usr/lib/i386-linux-gnu
ec2-user@ip-10-236-190-141:~$ sudo ln -s libGLEW.so.1.6 libGLEW.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libGLEWmx.so.1.6 libGLEWmx.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libXi.so.6 libXi.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libX11.so.6 libX11.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libXrandr.so.2 libXrandr.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libXxf86vm.so.1 libXxf86vm.so
ec2-user@ip-10-236-190-141:~$ sudo ln -s libstdc++.so.5 libstdc++.so

# Repeat to create symbolic links for x86_64
ec2-user@ip-10-236-190-141:~$ cd /usr/lib/x86_64-linux-gnu
```

Step 9: Remotely Connect to the Linux AMI (NVIDIA driver installed)

Configuring NVIDIA Linux Driver

First check to make sure the NVIDIA GPU is present using `lspci` to get the PCI-e bus ID of a K520.

```
ec2-user@ip-10-236-190-141:~$ lspci | grep NVIDIA
00:03.0 VGA compatible controller: NVIDIA Corporation GK104GL [GRID K520]
(rev a1)
ec2-user@ip-10-236-190-141:~/GRID/drivers$ sudo nvidia-xconfig --
busid=PCI:0:3:0 --use-display-device=None --virtual=1280x720
```

Take care of “`busid`”, the result of “`lspci`” is in hexadecimal but “`nvidia-xconfig`” needs a decimal. The “`--virtual`” option assigns the resolution of the virtual screen, you can set it as any other values you like. Check the **Device** section and **Screen** section of `/etc/X11/xorg.conf` in the table below. Refer to table with the highlighted lines :

```
Section "Device"
    Identifier      "Device0"
    Driver          "nvidia"
    VendorName      "NVIDIA Corporation"
    BusID           "PCI:0:3:0"
EndSection
Section "Screen"
    Identifier      "Screen0"
    Device          "Device0"
    Monitor         "Headless"
    DefaultDepth    24
    Option          "UseDisplayDevice" "None"
    SubSection      "Display"
        Virtual     1280 1024
        Depth       24
    EndSubSection
EndSection
```

Use `sudo vim /etc/X11/xorg.conf` to edit the file and delete all monitor sections, and add following monitor section and save this file with “`:wq!`” .

```
Section "Monitor"
    Identifier      "Headless"
    HorizSync       80.0 - 80.0
    VertRefresh     75.0
    Modeline        "1280x1024_75.00" 138.45 1280 1368 1504 1728 1024 1025
    1028 1069 -Hsync +Vsync
EndSection
```

Connect to the Amazon Linux AMI (CentOS) with `x11vnc`

Stop the X-Server. Assuming connection is via a SSH terminal. You need to be **root** or use **sudo**:

```
ec2-user@ip-10-236-190-141:~$ sudo init 3
ec2-user@ip-10-236-190-141:~$ sudo pkill X
ec2-user@ip-10-236-190-141:~$ sudo yum install x11vnc
ec2-user@ip-10-236-190-141:~$ sudo groupinstall Desktop
```

We next want to setup x11vnc for the actual remote graphics sessions. x11vnc provides much better graphics performance and is able to work with real X server displays that correspond to monitors, keyboards, and mice.

First we want to download the latest package (64-bit), you need to be **root**. This will download, install and start the x11vnc process

```
ec2-user@ip-10-236-190-141:~$ wget
http://dag.wieers.com/rpm/packages/x11vnc/x11vnc-0.9.13-1.el6.rf.x86_64.rpm
ec2-user@ip-10-236-190-141:~$ sudo install x11vnc-0.9.13-1.el6.rf.x86_64.rpm
```

From your local Linux/Mac client. We need to create a SSH tunnel between your client, and the SSH Linux AMI, while mapping the VNC port 5900 to port 5902 on your local client port. You can determine the local IP address of the Linux AMI by looking at the IP address next to the bash shell, during a SSH session. `<ipaddr_local>` = IP address of the Linux AMI as seen on the EC2 internal network (in this case it is 10.236.190.141). `<ipaddr_public>` = Publically accessible IP address of the Linux AMI with information provided from the EC2 console.

```
ssh -L <port_client>:<ipaddr_local>:<port_server> userid@<ipaddr_public> -i
{path}/keypair.pem
```

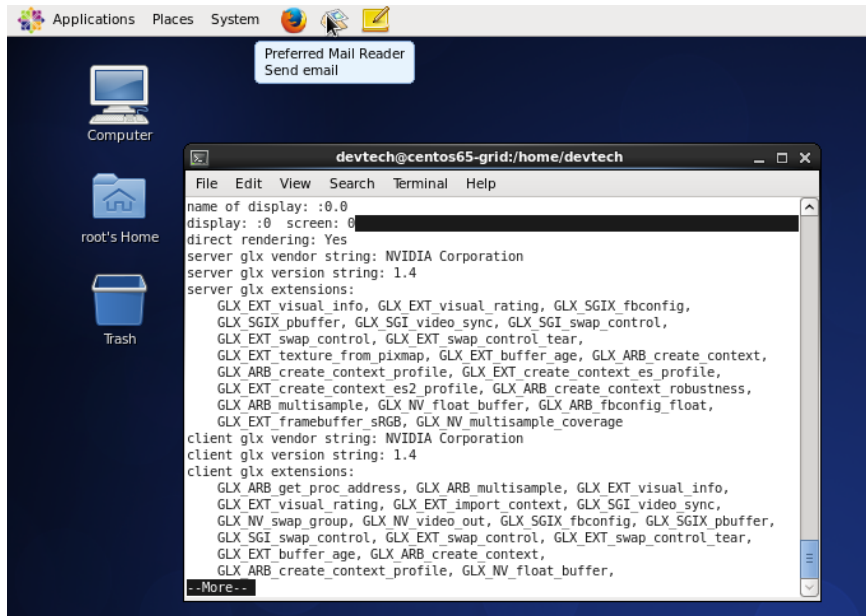
```
ec2-user@ip-10-236-190-141:~$ ssh -L 5902:10.236.190.141:5900
ec2-user@<ipaddr_public> -i {path}/keypair.pem
```

Start the X server and make the current **DISPLAY=:1**, and then start the x11vnc server.

```
ec2-user@ip-10-236-190-141:~$ sudo X &
ec2-user@ip-10-236-190-141:~$ export DISPLAY=:1
ec2-user@ip-10-236-190-141:~$ x11vnc -display :1 &
ec2-user@ip-10-236-190-141:~$ xterm &
```

Find the address and port number provided by **x11vnc**. It will be shown under **"HOSTNAME:<portnumber>"**. Now to connect to the x11vnc server, you can use **vncviewer** or RealVNC to **"IP address:<portnumber>"**. The default port is **5900**.

In the X-server VNC session, open a terminal session and run **glxinfo**. Confirm that **“NVIDIA Corporation”** is displayed twice under **“glx vendor string”**. You can now run the NVIDIA Capture SDK samples.



The screenshot shows a VNC session interface with a desktop background. On the left, there are icons for 'Computer', 'root's Home', and 'Trash'. At the top, there are menu bars for 'Applications', 'Places', and 'System'. A 'Preferred Mail Reader' button is visible. The central terminal window, titled 'devtech@centos65-grid:/home/devtech', displays the output of the 'glxinfo' command. The output includes display information, server and client GLX vendor strings (both 'NVIDIA Corporation'), and a list of GLX extensions. The text is as follows:

```
devtech@centos65-grid:/home/devtech
File Edit View Search Terminal Help
name of display: :0.0
display: :0 screen: 0
direct rendering: Yes
server glx vendor string: NVIDIA Corporation
server glx version string: 1.4
server glx extensions:
    GLX_EXT_visual_info, GLX_EXT_visual_rating, GLX_SGIX_fbconfig,
    GLX_SGIX_pbuffer, GLX_SGI_video_sync, GLX_SGI_swap_control,
    GLX_EXT_swap_control, GLX_EXT_swap_control_tear,
    GLX_EXT_texture_from_pixmap, GLX_EXT_buffer_age, GLX_ARB_create_context,
    GLX_ARB_create_context_profile, GLX_EXT_create_context_es_profile,
    GLX_EXT_create_context_es2_profile, GLX_ARB_create_context_robustness,
    GLX_ARB_multisample, GLX_NV_float_buffer, GLX_ARB_fbconfig_float,
    GLX_EXT_framebuffer_sRGB, GLX_NV_multisample_coverage
client glx vendor string: NVIDIA Corporation
client glx version string: 1.4
client glx extensions:
    GLX_ARB_get_proc_address, GLX_ARB_multisample, GLX_EXT_visual_info,
    GLX_EXT_visual_rating, GLX_EXT_import_context, GLX_SGI_video_sync,
    GLX_NV_swap_group, GLX_NV_video_out, GLX_SGIX_fbconfig, GLX_SGIX_pbuffer,
    GLX_SGI_swap_control, GLX_EXT_swap_control, GLX_EXT_swap_control_tear,
    GLX_EXT_buffer_age, GLX_ARB_create_context,
    GLX_ARB_create_context_profile, GLX_NV_float_buffer,
--More--
```

Configuring Linux AMI (Ubuntu 14.04 LTS with HVM)

From your local Linux machine, create a SSH tunnel for the purpose of VPN between the local machine and the SSH Linux AMI. Here we are going to tunnel into the the Linux AMI VNC port 5900 to y our local client machine via port 5902 . To determine the local IP address of the Linux AMI, it can be found by IP address at the bash prompt. <ipaddr_local> = IP address of the Linux AMI as seen on the EC2 internal network. <ipaddr_public> = Publically accessible IP address of the Linux AMI as seen from the EC2 console.

```
<port_client>:<ipaddr_local>           = 5902 is what we will use from localhost
<port_server> userid@<ipaddr_public>   = 5900 is what x11vnc will see
```

```
ubuntu@ip-10-236-190-141:~$ ssh -L 5902:10.236.190.141:5900
                           ubuntu@<ipaddr_public> -i {path}/keypair.pem
```

Stop the X-Server. You need to be **root** or use **sudo**:

```
ubuntu@ip-10-236-190-141:~$ sudo service lightdm stop
ubuntu@ip-10-236-190-141:~$ sudo pkill X
```

Now we want to run **dpkg-reconfigure** to allow **Any User** to start up the X server without having to be sudo. When prompted by the tool, choose **Anyone**.

```
ubuntu@ip-10-236-190-141:~$ sudo dpkg-reconfigure x11-common
ubuntu@ip-10-236-190-141:~$ X &
ubuntu@ip-10-236-190-141:~$ export DISPLAY=:1
ubuntu@ip-10-236-190-141:~$ x11vnc -display :1 &
ubuntu@ip-10-236-190-141:~$ unity &
```

Now remotely log into the Linux AMI from VNC, specifying the port want to run. From VNC dialog or **vncviewer** command line client, enter **localhost:<port_client>** to remotely connect via VNC into the Linux AMI and use the NVIDIA Graphics GPU. In order to confirm that we are running the NVIDIA GPU driver, enter this at the prompt.

```
ubuntu@ip-10-236-190-141:~$ vncviewer localhost:5902
ubuntu@ip-10-236-190-141:~$ glxinfo | more
```

Change to the NVIDIA Capture SDK folder. From here you can make the samples and run them

```
cd {NVIDIACaptureSDK_Installation}\Samples
make
```

Optional: Installing the Amazon EC2 Command Line Interface Tools on Windows

You can skip this part if you only want to use EC2 console to launch the instance.

The Amazon EC2 command line interface tools (also called the CLI tools) wrap the Amazon EC2 API actions. These tools are written in Java and include shell scripts for both Windows and Linux/UNIX/MacOSX.

Complete the following tasks to set up Amazon EC2 environment:

1. Download the CLI Tools
2. Set the JAVA_HOME Environment Variable
3. Set the EC2_HOME Environment Variable
4. Set the EC2_URL Environment Variable
5. Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables
6. (Optional) Set the Region
7. (Optional) Use a Proxy
8. Download Remote Desktop

Step 1: Download the Command Line Interface Tools (CLI Tools)

The AWS Command Line Interface (CLI) is a unified tool to manage AWS services.

- Requires [Python](#) 2.6 or higher.
- [pip](#) or [easy install](#)

To install aws services using pip, run the following command:

[pip install awscli](#)

To install aws services using [easy install](#), run the following command:

[easy_install awscli](#)

The CLI tools are available as a ZIP file on this site: [Amazon EC2 CLI Tools](#). The tools are written in Java and include shell scripts for both Windows and Linux/UNIX/Mac OSX. The ZIP file is self-contained, no installation is required. Simply download the file and unzip it.

Step 2: Setup the JAVA_HOME Environment Variable

The Amazon EC2 CLI tools require Java. They read the JAVA_HOME environment variable to locate the Java runtime. This environment variable should specify the full path of the directory that contains a subdirectory named bin that contains the Java executable you installed (`java.exe`).

To set the JAVA_HOME environment variable on your computer or instance

1. If you don't have Java 1.6 or later installed, download and install Java. Either a JRE or JDK installation is acceptable. To view and download JREs for a range of platforms, see [Free Java Download](#).
2. Set JAVA_HOME to the full path of the Java home directory. For example, if Java executable is in `C:\Program Files (x86)\Java\jre7\bin`, set JAVA_HOME to `C:\Program Files (x86)\Java\jre7`.

Important

These steps don't update the environment variables in the current Command Prompt windows. The Command Prompt windows that is open after completing these steps will contain the updates. This is why it's necessary for you to open a new Command Prompt window to verify that environment is set up properly.

- a. Click **Start**, right-click **Computer**, and then click **Properties**.
- b. Click **Advanced system settings**.
- c. Click **Environment Variables**.
- d. Under **System variables**, click **New**.
- e. In **Variable name**, type **JAVA_HOME**.
- f. In **Variable value**, type the path to of Java home directory (enclosing the path in quotation marks if the path contains spaces). For example,

"C:\Program Files (x86)\Java\jre7"

Important

Don't include the bin directory in JAVA_HOME. This is a common mistake, but the CLI tools won't work if this is done.

- g. Click **OK**.

3. Open a new Command Prompt window and verify the JAVA_HOME setting using this command.

```
C:\> %JAVA_HOME%\bin\java -version
```

If the environment variable is set correctly, the output looks something like this.

```
java version "1.7.0_05"  
Java(TM) SE Runtime Environment (build 1.7.0_05-b05)  
Java HotSpot(TM) Client VM (build 23.1-b03, mixed mode, sharing)
```

Otherwise, check the setting of JAVA_HOME, fix any errors, open a new Command Prompt window, and try the command again.

4. Add the `bin` directory that contains the Java executable to your path before other versions of Java.
 - a. In **System variables**, select **Path**, and then click **Edit**.
 - b. In **Variable values**, before any other versions of Java add `;%JAVA_HOME%\bin`.
5. Open a new Command Prompt window and verify the update version of the `Path` environment variable using this command.

```
C:\> java -version
```

You should see the same output as before. Otherwise, check the setting of Path, fix any errors, open a new Command Prompt window, and try the command again.

Step 3: Setup the EC2_HOME Environment Variable

The Amazon EC2 CLI tools read the **EC2_HOME** environment variable to locate supporting libraries. It is required to set this environment variable to the path where the CLI tools are unzipped. This directory is named **EC2-api-tools-w.x.y.z** (where w, x, y, and z are components of the version number). It contains sub-directories named `bin` and `lib`.

To set the EC2_HOME environment variable on your computer or instance

1. Set **EC2_HOME** to the path of the directory into which you unzipped the CLI tools.

Important

These steps don't update the environment variables in the current Command Prompt windows. After completing these steps, open a new Command Prompt and these updates will be available and verify the environment variables are set.

- a. Click **Start**, right-click **Computer**, and then click **Properties**.
 - b. Click **Advanced system settings**.
 - c. Click **Environment Variables**.
 - d. Under **System variables**, click **New**.
 - e. In **Variable name**, type **EC2_HOME**.
 - f. In **Variable value**, type the path to the directory where you installed the CLI tools.
For example, **C:\AWS\EC2\EC2-api-tools-1.6.7.4**.
2. Open a new Command Prompt window and verify your EC2_HOME setting using this command.

```
C:\> dir "%EC2_HOME%"
```

If the environment variable is set correctly, output for the directory listing can be seen. If there is a File Not Found error, check the setting of EC2_HOME, fix any errors, open a new Command Prompt window, and try the command again.

3. Add the `bin` directory for the tools to your system `Path` environment variable. The rest of this guide assumes that you've done this.

Path environment can be updated as follows:

- a. In **System variables**, select **Path**, and then click **Edit**.
- b. In **Variable values**, add `%EC2_HOME%\bin`.

Step 4: Setup the EC2_URL Environment Variable

Amazon EC2 CLI tools read the **EC2_URL** environment variable to locate the endpoint. An endpoint is a URL that is the entry point for a web service.

1. Set **EC2_URL** to the <https://ec2.amazonaws.com> such that the API tools will point to this endpoint.
2. An alternative to setting **EC2_URL**. Use the **-U** flag to specify this endpoint.

Example: **EC2-run-instances ami-2bafd842 -U <https://ec2.amazonaws.com>**

Step 5: Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables

Access keys identify the Amazon EC2 CLI tools. There are two types of access keys: *access key IDs* and *secret access keys*. You should store your access keys in a safe place after creation. Although you the access key ID can be retrieved from the [Your Security Credentials](#) page, the secret access key is not available on this page. If you lose your secret access key, you will need to create a new access key before you can use the CLI tools again.

Every time you issue a command, you must specify your access keys using the **--aws-access-key** and **--aws-secret-key** (or **-O** and **-W**) options. Alternatively, you might find it easier to store your access keys using the following environment variables:

- **AWS_ACCESS_KEY**—Your access key ID
- **AWS_SECRET_KEY**—Your secret access key

If these environment variables are set properly, their values serve as the default values for these required options, so you can omit them from the command line.

The following procedure describes how to create environment variables that specify your access keys.

To set up environment variables on our computer or instance

1. Click **Start**, right-click **Computer**, and then click **Properties**.
2. Click **Advanced system settings**.
3. Click **Environment Variables**.
4. Under **System variables**, click **New**.
5. In **Variable name**, type **AWS_ACCESS_KEY**.
6. In **Variable value**, specify your access key ID.
7. Under **System variables**, click **New**.
8. In **Variable name**, type **AWS_SECRET_KEY**.
9. In **Variable value**, specify your secret access key.

To verify that all environment variables are set up correctly, open a new Command Prompt window and run the following command.

```
C:\> EC2-describe-regions
```

If your environment variables are set correctly, you'll see output that looks something like this.

```
REGION  us-east-1      EC2.us-east-1.amazonaws.com
REGION  eu-west-1      EC2.eu-west-1.amazonaws.com
REGION  sa-east-1      EC2.sa-east-1.amazonaws.com
REGION  ap-northeast-1 EC2.ap-northeast-1.amazonaws.com
REGION  us-west-2      EC2.us-west-2.amazonaws.com
REGION  us-west-1      EC2.us-west-1.amazonaws.com
REGION  ap-southeast-1 EC2.ap-southeast-1.amazonaws.com
```

If you get an error that this command is not recognized as an internal or external command, check the setting of `Path`, fix any errors, open a new Command Prompt window, and try the command again.

If you get an error that required option **-O** is missing, check the setting of `AWS_ACCESS_KEY`, fix any errors, open a new Command Prompt window, and try the command again.

If you get an error that required option **-W** is missing, check the setting of `AWS_SECRET_KEY`, fix any errors, open a new Command Prompt window, and try the command again.

Step 6: Set the Region (Optional)

By default, the Amazon EC2 CLI tools use the **us-east-1** region with the **EC2.us-east-1.amazonaws.com** service endpoint URL. If your instances are in a different region, you must specify the region where your instances reside. For example, if your instances are in Europe, you must specify the **eu-west-1** region by using the **--region eu-west-1** option or by setting the **EC2_URL** environment variable.

The next section describes how to specify a different region by changing the service endpoint URL.

To specify a different region on your computer or instance

1. To view available regions, see [Regions and Endpoints](#) in the *Amazon Web Services General Reference*.
2. To change the service endpoint, set the **EC2_URL** environment variable.

The following example sets **EC2_URL**.

- a. Click **Start**, right-click **Computer**, and then click **Properties**.
- b. Click **Advanced system settings**.
- c. Click **Environment Variables**.
- d. Under **System variables**, click **New**.
- e. In **Variable name**, type **EC2_URL**.
- f. In **Variable value**, type **https://EC2.us-east-1.amazonaws.com**.

Step 7: Use a Proxy (Optional)

If the computer you have installed the CLI tools on requires the use of a proxy server, you must tell the CLI tools to use the proxy server with the `EC2_JVM_ARGS` environment variable.

The following table contains the proxy configuration properties that can be set for the `EC2_JVM_ARGS` variable. The properties that are required will depend on the type of proxy server being used. For example, the `http.proxyDomain` and `http.proxyWorkstation` properties are only used with a Windows NTLM proxy.

Property	Description
<code>https.proxyHost</code>	HTTPS proxy host. Use when <code>EC2_URL</code> specifies an HTTPS host.
<code>https.proxyPort</code>	HTTPS proxy port. Use when <code>EC2_URL</code> specifies an HTTPS host.
<code>http.proxyHost</code>	HTTP proxy host. Use when <code>EC2_URL</code> specifies an HTTP host.
<code>http.proxyPort</code>	HTTP proxy port. Use when <code>EC2_URL</code> specifies an HTTP host.
<code>http.proxyDomain</code>	Proxy domain (HTTPS and HTTP)
<code>http.proxyWorkstation</code>	Proxy workstation (HTTPS and HTTP)
<code>http.proxyUser</code>	Proxy user name (HTTPS and HTTP)
<code>http.proxyPass</code>	Proxy password (HTTPS and HTTP)
<code>http.nonProxyHosts</code>	A list of hosts that should be reached directly, bypassing the proxy. Each item in the list is separated by ' '.

To set up the `EC2_JVM_ARGS` environment variable on your computer or instance

1. Click **Start**, right-click **Computer**, and then click **Properties**.
2. Click **Advanced system settings**.
3. Click **Environment Variables**.
4. Under **System variables**, click **New**.
5. In **Variable name**, type `EC2_JVM_ARGS`.
6. In **Variable value**, specify the proxy configuration properties. For example,
`"-Dhttps.proxyHost=my.proxy.com -Dhttps.proxyPort=8080"`.

Working with instances

Amazon EC2 API tools provide descriptions, syntax, and usage examples for each of the commands for EC2 and Amazon Virtual Private Cloud (VPC).

Below are the most commonly used command line settings. For the full list of command line tools, [click here](#).

1. Create key-pair
2. Create security group
3. Create Instances
4. Start Instances
5. Stop Instances
6. Reboot Instances
7. Terminate Instances

Create a key-pair

Creates a 2048-bit RSA key pair with the specified name. Amazon EC2 stores the public key and displays the private key for you to save to a file. The private key is returned as an unencrypted PEM encoded PKCS#8 private key. If a key with the specified name already exists, Amazon EC2 returns an error.

Syntax: `ec2-create-keypair` **key**

Example: `ec2-create-keypair my-key-pair`

This example command creates a key pair named my-key-pair.

```
PROMPT> ec2-create-keypair my-key-pair

KEYPAIR    my-key-pair          1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
----- BEGIN RSA PRIVATE KEY -----
MIICiTCcAfiCCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xZDASBgNVBAStC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHZAAd
BgkqhkiG9w0BCQEWEG5vb25lQGFTYXpvi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xZDASBgNVBAStC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHZAAdBgkqhkiG9w0BCQEWEG5vb25lQGFT
YXpvi5jb20wG9z8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLyGVik60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJl1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE
```

```
-----END RSA PRIVATE KEY-----
```

Create a file named `my-key-pair.pem` and paste the entire key from the response into this file, including the following lines.

```
"----- BEGIN RSA PRIVATE KEY -----"  
"-----END RSA PRIVATE KEY-----"
```

Confirm that the file contents are similar to the following and save the file to a local directory.

```
----- BEGIN RSA PRIVATE KEY -----  
MIICiTCcAfICcQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC  
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6  
b24xFDASBgNVBAStC01BTsBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxH2Ad  
BgkqhkiG9w0BCQEWEG5vb25lQGgtYXpvi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN  
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD  
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAStC01BTsBDb25z  
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxH2AdBgkqhkiG9w0BCQEWEG5vb25lQGgt  
YXpvi5jb20wZGZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ  
21uUSfwfEvySWtC2XADZ4nB+BLygVik60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T  
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE  
Ibb3OhjZnzcvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4  
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb  
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvJx79LjSTb  
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE  
-----END RSA PRIVATE KEY-----
```

Keep this file in a safe place; it is required to decrypt login information when you connect to an instance that you launched using this key pair.

If you're using an SSH client on a Linux computer to connect to your instance, use the following command to set the permissions of your private key file so that only you can read it.

```
$ chmod 400 my-key-pair.pem
```

Creating a Security Group

Syntax: `ec2-create-group group_name -d description [-c vpc_id]`

Example: `ec2-create-group my-w2k8r2 -d "Windows Server 2008 R2 Group"`

Add rules for security group

You can add rules for your security groups using the `ec2-authorize`.

Syntax: `ec2-authorize group [--egress] [-P protocol]
 (-p port_range | -t icmp_type_code)
 [-u source_or_dest_group_owner ...]
 [-o source_or_dest_group ...]
 [-s source_or_dest_cidr ...]`

Example:

This example command grants TCP port 22 (SSH) access for all address range to the security group named `my-w2k8r2`.

```
PROMPT> ec2-authorize my-w2k8r2 -P tcp -p 22 -s 0.0.0.0/0
GROUP          my-w2k8r2
PERMISSION     my-w2k8r2  ALLOWS      tcp 22 22  FROM  CIDR  0.0.0.0/0
ingress
```

These ports need to be opened when you connect to your instance:

- a. SSH: TCP port 22
- b. RDP: TCP port 3389

Create Instances

Launches a specified number of instances of an AMI for which you have permissions. If Amazon EC2 cannot launch the minimum number of instances requested in a single Availability Zone, no instances are launched.

Note

Every instance is launched in a security group (which you create using the `EC2-create-group` command). If you don't specify a security group at launch time, the default security group is used.

- Specify “g2.xlarge” as the instance type for the `RunInstances` call.
- AMI id can be found from [AWS marketplace](#). Here is a list copy from marketplace. It may be updated in the future.

- Windows Server 2008 R2

Region	ID
US East (Virginia)	ami-bd6a38d4
US West (Oregon)	ami-dc8019ec
US West (Northern California)	ami-b23007f7
EU West (Ireland)	ami-42759435

- Windows Server 2008

Region	ID
US East (Virginia)	ami-8d6a38e4
US West (Oregon)	ami-da8019ea
US West (Northern California)	ami-be3007fb
EU West (Ireland)	ami-5a75942d

- Amazon Linux

Region	ID
US East (Virginia)	ami-1b597c72
US West (Oregon)	ami-9a9f04aa
US West (Northern California)	ami-2cbe8869
EU West (Ireland)	ami-ab836edc

To create an instance id, you will run the following command to launch a Windows Server 2008 R2 instance in US east, assuming you’ve created a keypair called “`nvidia.pem`”, and a security group with the RDP ports open called “`my-w2k8r2`”:

```
EC2-run-instances ami-bd6a38d4 -t g2.2xlarge -g my-w2k8r2 -k nvidia
```

Starting Instances

This command starts an instance that uses an Amazon EBS volume as its root device.

Instances that use Amazon EBS volumes as their root devices can be quickly stopped and started. When an instance is stopped, the compute resources are released and you are not billed for hourly instance usage. However, root partition Amazon EBS volume remains, continues to persist your data, and charged for Amazon EBS volume usage. Each time you transition an instance from stopped to started, Amazon will charge a full instance hour, even if transitions happen multiple times within a single hour.

Syntax: **EC2-start-instances** *instance_id*

The short version of this command is **EC2start**.

Stopping Instances

This command stops an instance that uses an Amazon EBS volume as its root device.

Syntax: **EC2-stop-instances** *instance_id*

The short version of this command is **EC2stop**.

Rebooting Instances

This command requests a reboot of one or more instances. This operation is asynchronous; it only queues a request to reboot the specified instances. The operation will succeed if the instances are valid and belong to you. Requests to reboot terminated instances are ignored.

Syntax: **EC2-reboot-instances** *instance_id*

The short version of this command is **EC2reboot**.

Terminating Instances

This command shuts down one or more instances. This operation is idempotent; if you terminate an instance more than once, each call succeeds.

Terminated instances will remain visible after termination (approximately one hour).

Syntax: **EC2-terminate-instances** *instance_id*

The short version of this command is **EC2kill**.

Connecting to Instances

When you start your AWS instance, you can use **ec2-describe-instances** to get instance's information.

Syntax: **EC2-describe-instances** *instance_id*

Example: **EC2-describe-instances** *i-94cff343*

The instance **i-94cff343** uses Windows Server 2008 R2, binding to security group **my-w2k8r2**, using keypair **nvidia.pem**. We get the below output:

```
PROMPT> ec2-describe-instances i-94cff343
RESERVATION      r-94cff343      676255659926      my-w2k8r2
INSTANCE i-94cff343      ami-bd6a38d4      ec2-50-19-36-251.compute-1.amazonaws.com
      ip-10-51-130-156.ec2.internal      running nvidia 0
[marketplace: crsfeeo2ey3gm102u2zy4q3z2]      g2.2xlarge      2013-12-06T05:25:11+0000
      us-east-1b      windows monitoring-disabled50.19.36.251
      10.51.130.156      ebs hvm xen
      df8f6f91-157e-4b49-ba5e-2c054019e295 sg-777e201c      default false
BLOCKDEVICE      /dev/sda1 vol-d5ca9ba2      2013-12-03T08:49:48.000Z true
```

To connect the instance with Remote Desktop, we need to obtain the password for this remote instance.

Syntax: **EC2-get-password** *instance_id* -k *key_file*

Example : **EC2-get-password** *i-94cff343* -k *nvidia.pem*

This will provide you the Windows password so you can remote into the Windows instance. This command will produce a result roughly 10-15 minutes after your run-instances command.

You can now connect to this instance with public DNS `ec2-50-19-36-251.compute-1.amazonaws.com` by remote desktop, user name is `\Administrator` and password is the one got from **EC2-get-password**.

[Or you can connect to the instance with TeamViewer/SSH.](#)

Notice

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication of otherwise under any patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all other information previously supplied. NVIDIA Corporation products are not authorized as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

Trademarks

NVIDIA and the NVIDIA logo are trademarks or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2011-2016 NVIDIA Corporation. All rights reserved.