

This paper was originally presented at the Southwest Fox conference in Gilbert, Arizona in October, 2012. <http://www.swfox.net>



Amazon Elastic Cloud Computing for Fun and Profit

*Rick Borup
Information Technology Associates
701 Devonshire Dr, Suite 127
Champaign, IL 61820
Voice: (217) 359-0918
Email: rborup@ita-software.com*

Gone are the days where banks of dedicated physical servers lined the floor, shelves, or racks in your hardware closet. If you're deploying Web-based solutions these days, you're most likely going to use a cloud-based virtual private server from one of the big names in the business, and they don't get much bigger than Amazon Elastic Cloud Computing (EC2) from Amazon Web Services (AWS). Getting started with EC2 can be a little intimidating, but once you learn the lingo and get used to the interface, you'll be amazed at the power and flexibility EC2 solutions provide. Come to this session and learn how to make EC2 work for you!

Table of Contents

Getting Started with Amazon Web Services	4
What is AWS	4
Creating an AWS account	4
Pay as you go	11
Getting started for Windows developers	11
EC2 AMIs, instance types, and pricing	12
Amazon machine instances (AMIs)	12
Instance types	12
Pricing	13
Creating and configuring a Windows server instance	14
Launching a new instance	16
Exploring your new server instance	19
EBS Volumes	21
Security groups	21
Connecting to your server	24
Working with your server instance	30
The EC2 Management Console	30
The Console Dashboard	30
My Instances	30
Stopping a running instance	31
Starting a stopped instance	32
Obtaining a static IP address	33
Rebooting your server	37
Backing up your server	37
Terminating your server instance	38
Termination protection	39
Leveraging your setup	39
Snapshots	40
Creating your own AMI	41
Notes from the field	44
Using FTP	44

Sending Email	45
Installing additional Windows features	45
Installing Windows updates.....	46
Summary	46
Acknowledgements	46
Glossary	46
Resources	47

Getting Started with Amazon Web Services

What is AWS

Amazon Web Services (AWS) is a family of infrastructure and services for virtual computing, storage, and related activities in the cloud. One of these services is the Elastic Compute Cloud (EC2), popularly referred to as Elastic Cloud Computing. Another is Elastic Block Storage (EBS). One of the key concepts behind Amazon Web Services is scalability. Amazon refers to this as *elasticity*, a term that crops up in many of their acronyms and service names.

Some of the largest enterprise and big data applications in the world run on AWS, but also some of the smallest. This paper focuses on the small end of the scale, covering the basics of setting up, configuring, and managing a virtual server running Windows Server. This is the requirement perhaps most frequently encountered by independent and small-shop Windows developers who need a way to host and run their Web-based applications.

When you first start working with AWS, you may feel overwhelmed by its scope and complexity, as well as by the veritable jungle of acronyms and terminology you'll encounter – AWS, AMI, EBS, EC2, EIP, availability zones, instance types, reserved instances, and security groups, among others.¹ One of the objectives of this session is to help reduce that initial confusion by explaining what these things are and how to use them. Fear not: it will all fall into place as you begin to use and get familiar with AWS.

Creating an AWS account

To use AWS, you need to have an account with Amazon. Creating an Amazon Web Services account is easy – all you need is an email address and a credit card. If you already have an Amazon account you use for buying books and things from amazon.com, you can use the same account for AWS. If you don't already have an account, head over to the AWS home page at aws.amazon.com and click the Sign Up Now button to set one up.

¹ See the Glossary at the end of this paper for definitions of these and other acronyms and terminology.

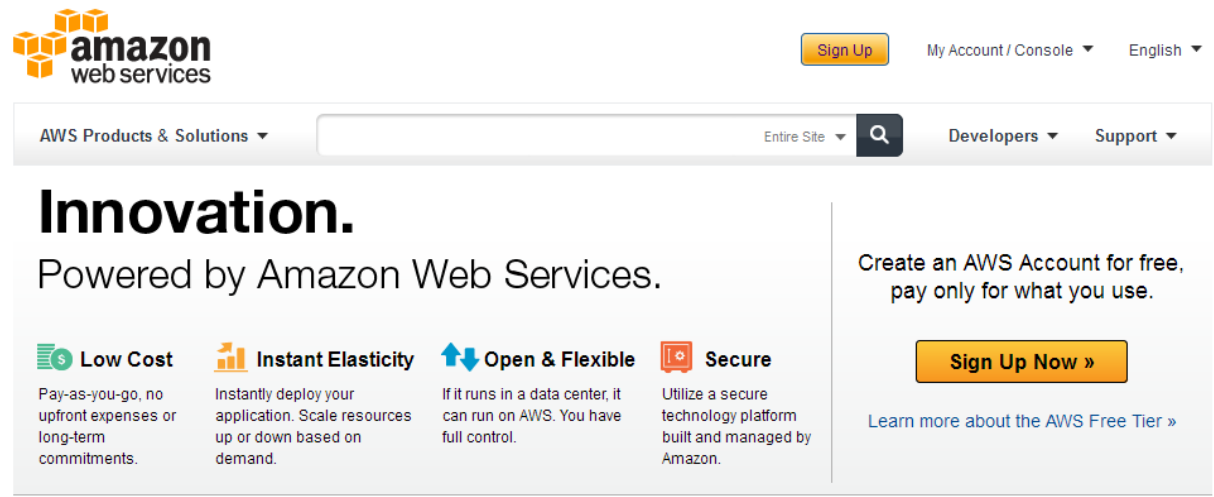


Figure 1: To create an account, go to aws.amazon.com and click the Sign Up Now button.

In the subsequent dialog, fill in your email address, select the *I am a new user* radio button and click *Sign in using our secure server*.



Figure 2: To begin the sign-up process, enter your email address, select the *I am a new user* button, and click the *Sign in using our secure server* button.

Before you begin the sign-up process, be ready to receive email at the address you want to use for your AWS account, and be sure you have access to a phone; part of the sign-up process requires you to receive a phone call and touch in a PIN number for identity verification. You'll also be asked to create a password for your AWS account, so you might want to have one in mind before you start. Finally, have your credit card handy because you'll need to enter it for AWS billing purposes.

Enter your name, email address, and create a password.

amazon web services™

Login Credentials

Use the form below to create login credentials that can be used for AWS as well as Amazon.com.

My name is: Rick Borup

My e-mail address is: [redacted]

Type it again: [redacted]

note: this is the e-mail address that we will use to contact you about your account

Enter a new password: [redacted]

Type it again: [redacted]

[Continue](#)

Figure 3: Establish your login credentials by providing your name, email address and selecting a password.

Enter your contact information and agree to the terms of service.

Contact Information

* required fields

Full Name*: Rick Borup

Company Name: Information Technology Assoc

Country*: United States

Address Line 1*: [redacted]
Street address, P.O. box, company name, c/o

Address Line 2: [redacted]
Apartment, suite, unit, building, floor, etc.

City*: Champaign

State, Province or Region*: IL

ZIP or Postal Code*: 61820

Phone number*: [redacted]

Security Check

Image: 

[Try a different image](#) [Why do we ask you to type these characters?](#)

Type the characters in the above image*: WJR39N

[Having Trouble? Contact us.](#)

AWS Customer Agreement

Check here to indicate that you have read and agree to the terms of the Amazon Web Services Customer Agreement. [\[2\]](#)

Figure 4: Enter your full name and contact information, and agree to the terms of service.

Enter your credit card information and billing address.

The screenshot shows the 'PAYMENT METHOD' step of the AWS account creation process. At the top, a progress bar indicates the steps: CREATE ACCOUNT, PAYMENT METHOD (current), IDENTITY VERIFICATION, and CONFIRMATION. A yellow box contains the text: 'Your AWS account credentials have been created, but in order to begin using any of the services, you will need to provide your payment information and continue. There is no fee to sign up and you only pay for what you use.' Below this is the section 'Enter Your Payment Information Below'. A paragraph explains that the credit card will not be charged until AWS usage begins and that charges will be billed to the provided card. A link for 'View detailed service pricing' is provided. The form includes a dropdown for 'Credit Card*' (set to 'Select card type'), a text input for 'Card Number*', a text input for 'Cardholder's Name*', and two dropdowns for 'Expiration Date*' (set to '01' and '2012'). Below the payment information is the section 'Enter Your Billing Address'. It asks to 'Select the billing address associated with your credit card.' and offers two radio button options: 'Use my contact address as my billing address' (selected) and 'Enter a new address'. The selected option shows the address '(701 Devonshire Dr, Suite 127, Champaign, IL 61820, US, (217) 359-0918)'. A 'Continue' button is at the bottom.

Figure 5: Enter your credit card information for billing purposes.

Enter the phone number you want Amazon to use for identity verification. Note that this does not have to be the same phone number you supplied with your contact information. When you're ready to receive the phone call, click the *Call Me Now* button to proceed to the next step.

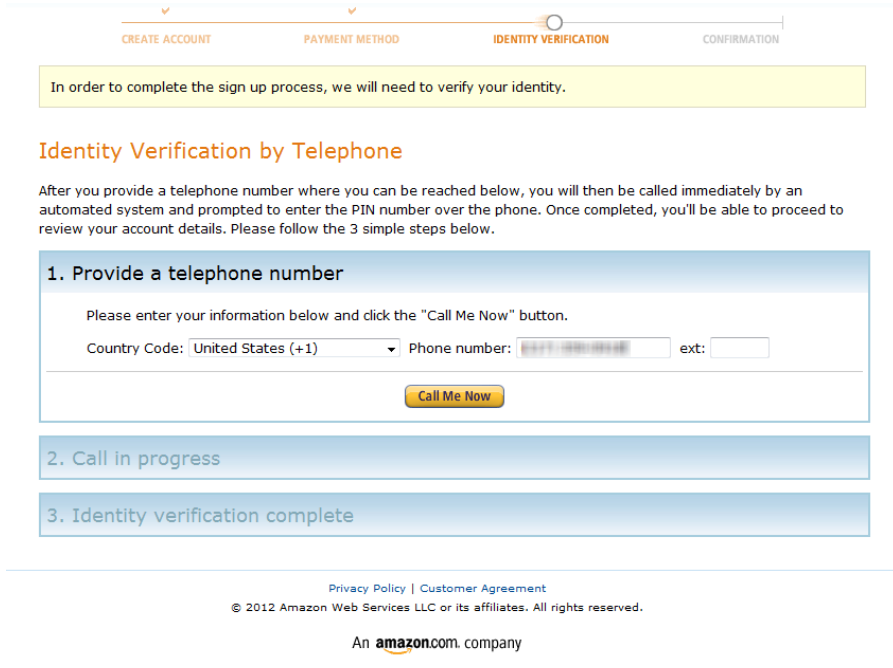


Figure 6: Provide the phone number you want Amazon to use for identity verification.

Amazon assigns you a PIN and initiates an automated phone call to the number you supplied.

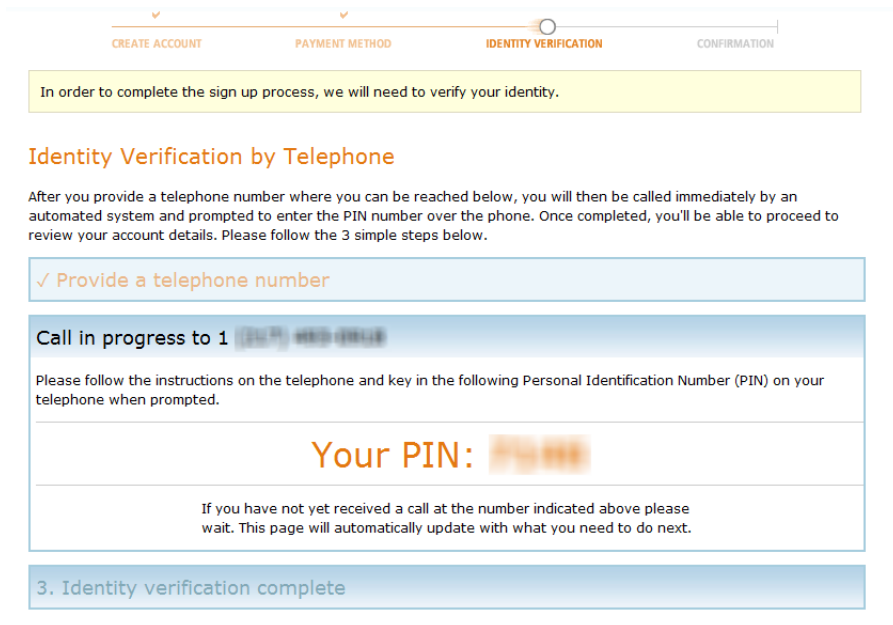


Figure 7: Amazon initiates an automated phone call to you and provides a PIN you'll be prompted to enter.

When you receive the phone call, you're prompted to touch in the PIN number assigned by Amazon. After you do this, the phone call terminates and the browser page is automatically refreshed to inform you your identity has been verified.

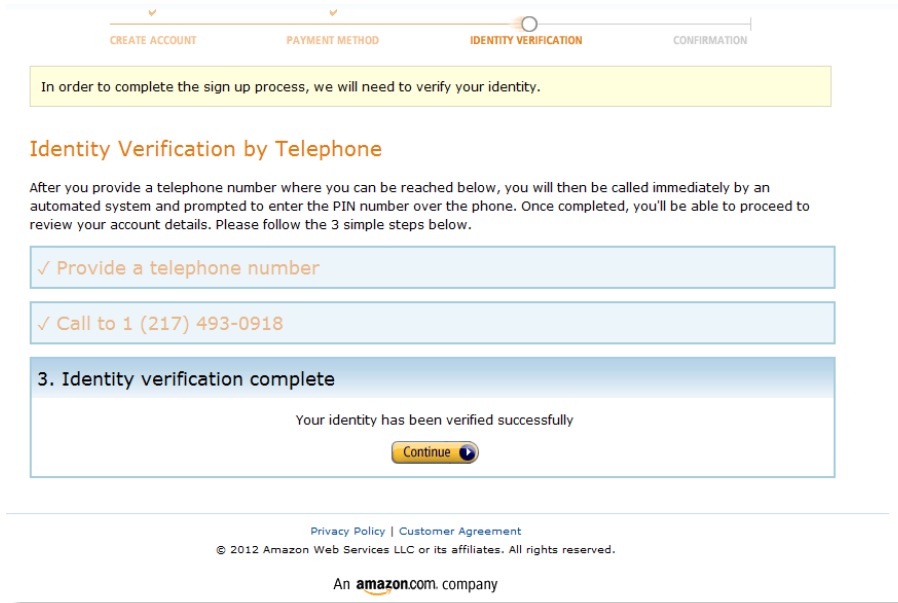


Figure 8: After entering your PIN number in response to the phone call, your identity verification is complete.

The sign-up process is now complete. When you click Continue, the web page informs you your account is being activated. You will receive an email when activation is complete – this takes normally only a few minutes.

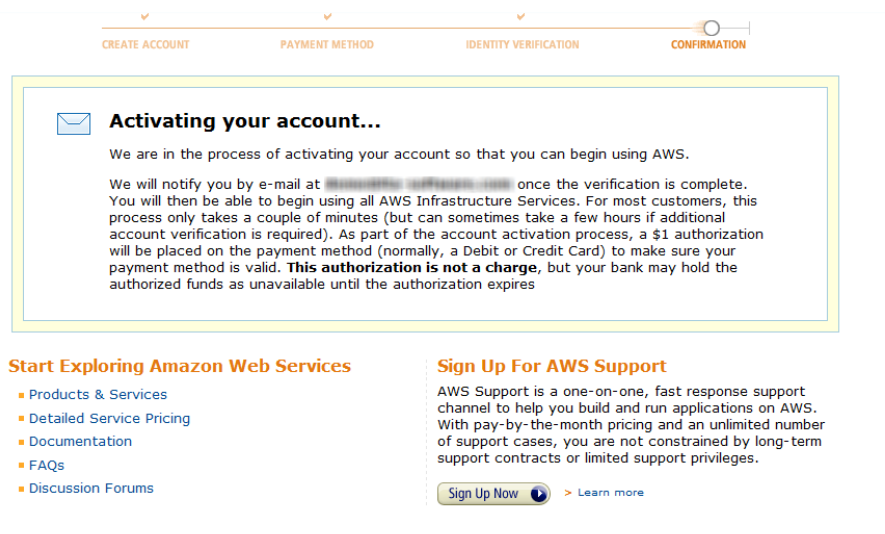


Figure 9: Your account gets activated as the final step in the sign-up process.

While you're waiting for your confirmation email to arrive, you might want to begin exploring some of the services, documentation, and support options available from AWS. As shown in Figure 9, the final page of the setup process has several helpful links for this purpose. It's a good idea to bookmark these links for later reference; the AWS documentation is easy to navigate but it's also quite extensive, so it's sometimes difficult to remember where you saw something you'd like to go back to later on.

When you receive the confirmation email, your account is activated and ready to go. The email also has several useful links to help you find your way around and get started using AWS. In particular, note the link to the AWS Management Console – it's the one you'll use most often to launch and manage your EC2 server instances and to access other resources.

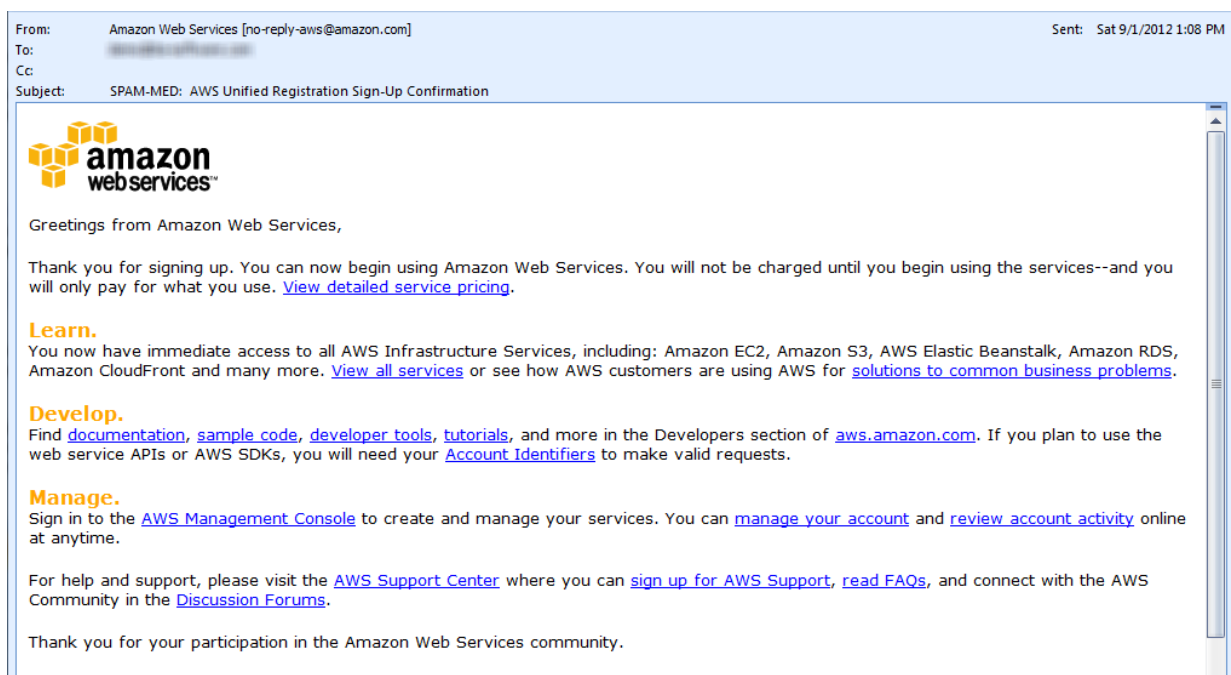


Figure10: The Sign-Up Confirmation from Amazon Web Services contains links to several useful resources. Note that my email provider identified this as medium-level spam – if your email provider is more aggressive in filtering spam than mine is, there's a chance you might not receive this email.

Finally, Amazon sends you a *Welcome to Amazon Web Services* email. One of the links in this email takes you to information about the free usage tier. This is worth checking out, because the free tier enables you to use certain types of AWS servers at no cost for a period of up to one year.

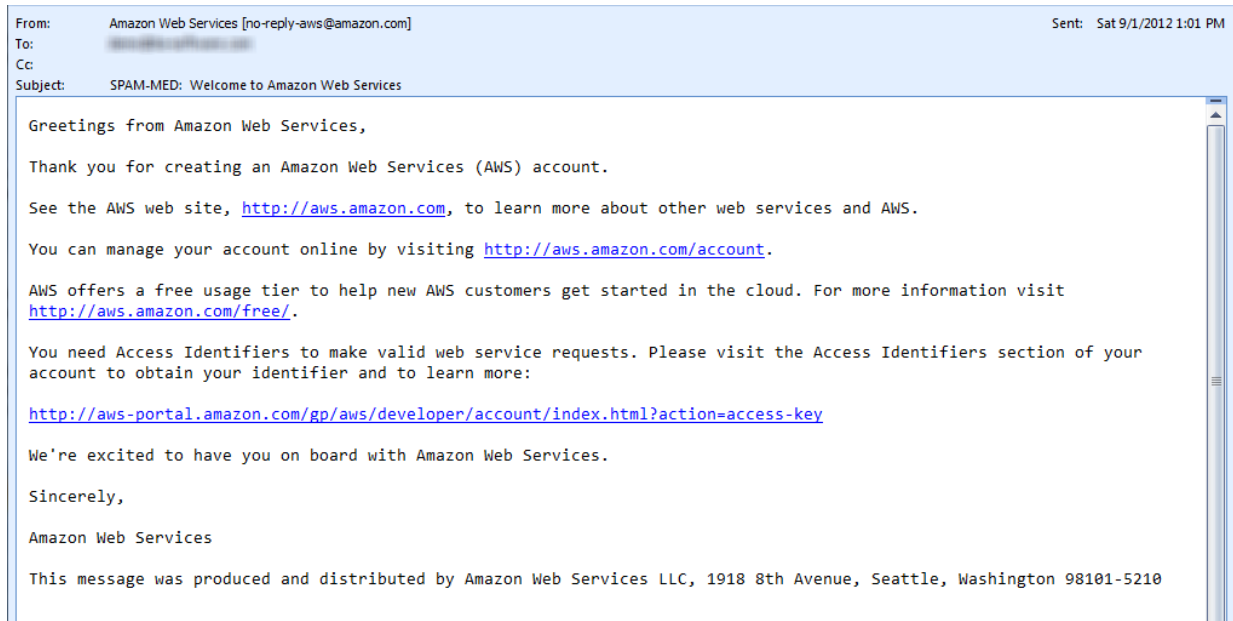


Figure 11: The Welcome email also provides some useful links to help get you started.

Pay as you go

Unless you contract for reserved instances or other pay-in-advance services, AWS is pay-as-you-go. In general, billing begins when you start a server instance and ends when you stop it.² The charge for a virtual server is typically in the range of pennies per hour, depending on the instance type. This means you can get started for a very small amount of money.

Amazon helps new AWS customers get started by offering a free usage tier. This enables new users to run a free Amazon EC2 Micro Instance and related services for a year. Micro Instances represent the lowest level of server instance available from AWS; they're also available on a paid basis outside of the free usage tier. A Micro Instance offers very limited computing power and storage resources and therefore may or may not be suitable for production use, but it's the most economical way to become familiar with how AWS works and how to launch, manage, and terminate server instances.

Getting started for Windows developers

If you're a Windows developer, one of the best ways to get started with AWS is to read the *Amazon Elastic Compute Cloud Microsoft Windows Guide*, which is available in the docs section of the AWS website.³ While I like online documents for quick lookups and easy reference, I prefer paper for front-to-back reading. When I was first getting started with

² Note that stopping an instance is not the same as terminating it; more on this later.

³ See the Resources section at the end of this paper for links to this and other valuable sources of information.

AWS, I found it worthwhile to print this document and have it spiral-bound at Kinko's. It runs about 100 pages. It's also a good online reference to keep handy even after you're comfortably up and running, so it's another one I'd recommend bookmarking in your browser.

EC2 AMIs, instance types, and pricing

Once you've set up your AWS account, you're ready to launch an instance. There are a couple of decisions you'll want to be ready to make before you jump right into this, though. The first is to choose the type of server you want to work with. There are several aspects to this choice, among them the operating system, the database software (if any), the application software (if any), and the instance type. Each of these has a bearing on the price you'll pay, so cost is also a consideration.

Amazon machine instances (AMIs)

Amazon EC2 provides pre-configured templates for many types of servers you can choose from. When you launch a server of a particular type, it's called an *instance* of that type, so these templates are called Amazon Machine Instances, or AMIs.

There are AMIs for Windows Server and also for various flavors of Linux. Within the AMIs for each OS you can choose various different feature sets, with each one being designed as the most suitable starting point for the particular role you want your server to fulfill. For example, there are AMIs designed for use as Web servers, as database servers, or as content management servers. Once your virtual server is up and running, you can of course install whatever software you want on it, but choosing the most suitable AMI as your starting point gives you a leg up.

Instance types

Amazon uses the term *instance type* to differentiate between the various levels of computing power and storage resources AWS allocates to your instance. Instance types are organized into categories, and each category comprises one or more instance types.

On the smallest end of the scale is the *micro instances* category, which has only one instance type. Micro instances come with the lowest level of CPU resources and lowest amount of available storage offered by AWS. Micro instances are great for learning because they're the least expensive. Although their capabilities are limited, micro instances can also be useful for production applications that have low CPU demands and low storage requirements.

In the middle of the scale is the *standard instances* category. In this category you'll find small, medium, large, and extra large instance types. These instance types differ according to the amount of memory, number of cores and number of EC2 Compute Units (ECUs), amount of disk storage, and processor architecture (32-bit or a 64-bit).

For applications with more demanding requirements, the high end of the scale includes categories offering high-memory, high-CPU, high-I/O, and cluster instance types.

Pricing

So what does all of this cost? In general, you pay only for what you use. The AWS pricing scheme can get pretty intricate, but the basic charge is per hour of usage. One AWS server running for one hour is called an *instance hour*. Servers running Linux are typically somewhat less expensive than those running Windows. Prices also vary a bit depending on the *region* you choose to host your server: U.S. East (Virginia), U.S. West (Northern California or Oregon), or a non-U.S. based location.

There is also a price difference based on whether you choose an *on-demand instance*, which has no up-front cost and no long-term commitment, or a *reserved instance*, which requires an up-front cost along with a commitment for a 1-year or a 3-year term in return for a reduced hourly price. Reserved instances are further broken down and priced according to the desired level of utilization – light, medium, or heavy. Light and medium utilization reserved instances are billed by the hour for the time the instances are in a running state, while heavy utilization reserved instances are billed for every hour, although at a lower rate, regardless of state.

Table 1 shows the current pricing⁴ for a micro instance compared to a small instance for both an on-demand and a light-utilization reserved instance, running Windows Server software and located in the U.S. East region. The monthly cost and annualized cost columns are based on my own calculations as explained in the footnotes below the table.

Table 1: A comparison of the average monthly price for typical small-scale Windows Server configurations.

Instance Type	On-Demand / Reserved	Price per hour	Monthly cost [§]	Annualized cost [†]
Micro	On-Demand	\$0.020	\$14.40	\$172.80
Micro	Reserved (\$23/yr)	\$0.018	\$12.96	\$178.52
Small	On-Demand	\$0.115	\$82.80	\$993.60
Small	Reserved (\$69/yr)	\$0.059	\$42.48	\$578.76

[§] Based on 720 hours per month

[†] Monthly cost times 12, plus the up-front cost for a 1-yr reserved instance at the price indicated

If you plan to host your application for an extended period of time and leave your server running 24/7, you can save money with a reserved instance. Table 1 reveals that a 1-year reserved small instance has a significant cost advantage over a corresponding on-demand

⁴ The prices shown here are from the pricing section of the AWS website at aws.amazon.com/ec2/#pricing and are current as of the date of this writing in August, 2012. Prices may change, so please visit that link for current pricing.

instance. The cost advantage of a reserved instance is greater if you lock in for a 3-year term rather than a 1-year term.

For a more complete picture, Chart 1 illustrates the costs of four types of small instances running Windows Server in the U.S. East region over a one-year period. The cost advantage of the three reserved instances (RL, RM, and RH) over the comparable on-demand instance (OD) kicks in somewhere in the first two or three months, depending on the level of utilization. The savings are significant by the end of the year, in the neighborhood of \$400 to \$500. Among the three reserved instance types, the cost advantage of the higher utilization instances compared to the lower ones doesn't kick in until the 7th or 8th month, but by the end of the year the high-utilization reserved instance is the least expensive, with the medium and low-utilization instances coming in second and third respectively.



Chart 1: Over the course of a year, a reserved instance saves you money compared to an on-demand instance. (OD=on-demand RL=reserved, low-utilization RM=reserved, medium-utilization RH=reserved, high-utilization, Calculations are by the author, based on prices quoted on the AWS Web site as of August, 2012.)

Creating and configuring a Windows server instance

When you sign in to your AWS account, the default start page is the Console Home page (see Figure 12). If you're already signed in, you can get to this page directly via its URL <https://console.aws.amazon.com/console/home>.

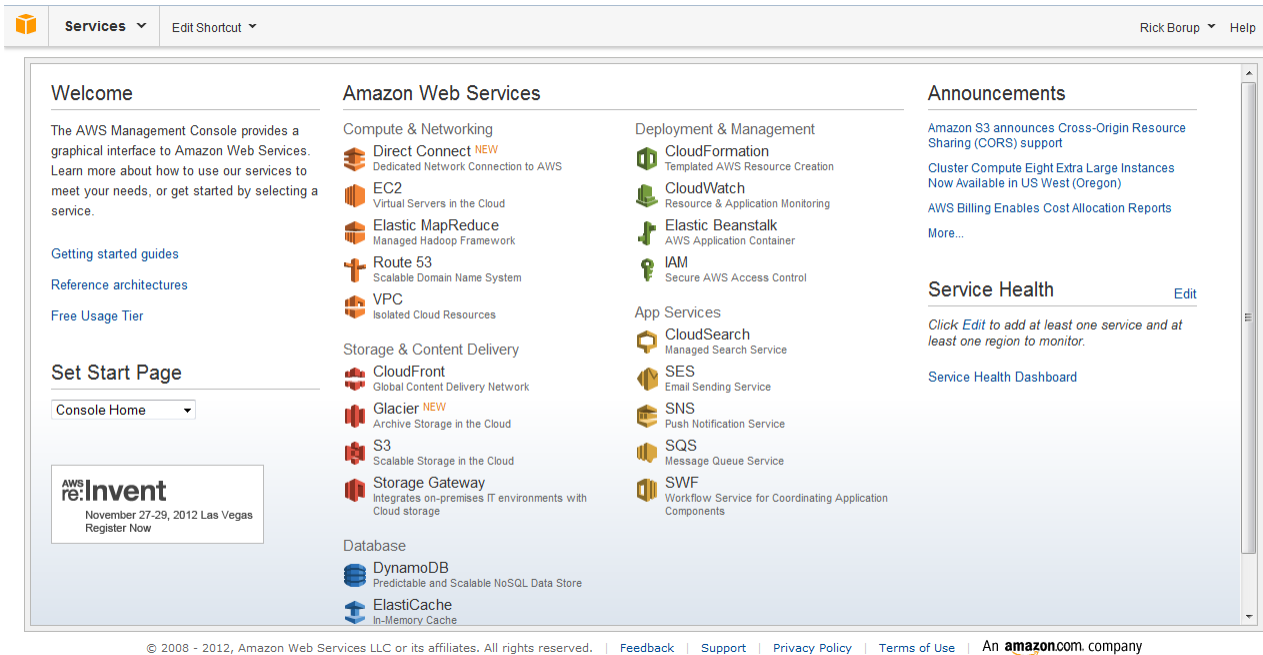


Figure 12: The Console Home page is the default start page when you sign in to AWS.

This Console Home page provides convenient links to various Amazon Web Services you can work with. The one we're interested in for this session is the EC2 service. Note the area on the left in Figure 12, where you can set a different start page if you want to. If you want to skip the home page and go directly to the EC2 Console Dashboard when you sign in, you can select EC2 from the Set Start Page drop-down list.

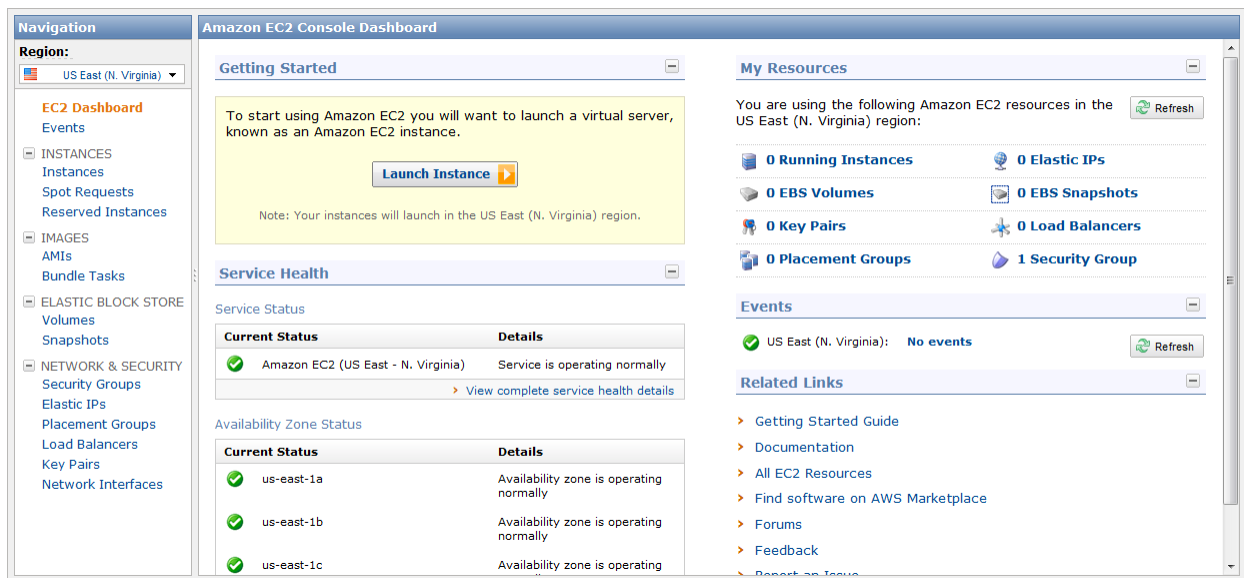


Figure 13: The EC2 Console Dashboard is the command center for launching and managing your EC2 instances.

The EC2 Console Dashboard is the command center for launching and managing your EC2 instances. The left side holds a navigation menu with links to the various categories of EC2 management. Figure 13 shows the dashboard as it appears when the *EC2 Dashboard* item is selected in the left-side navigation pane. Note the *My Resources* area in the upper right, which indicates no instances are currently running. This makes sense, because this is a new AWS account and no instances have been created yet.

Launching a new instance

Once you are logged in to your AWS account and have navigated to the EC2 Console Dashboard as shown in Figure 13, click the Launch Instance button to initiate the process of creating a new instance. This process begins by giving you a choice between the Classic Wizard, the Quick Launch Wizard, and the AWS Marketplace. The Quick Launch Wizard is probably the easiest way to get started.

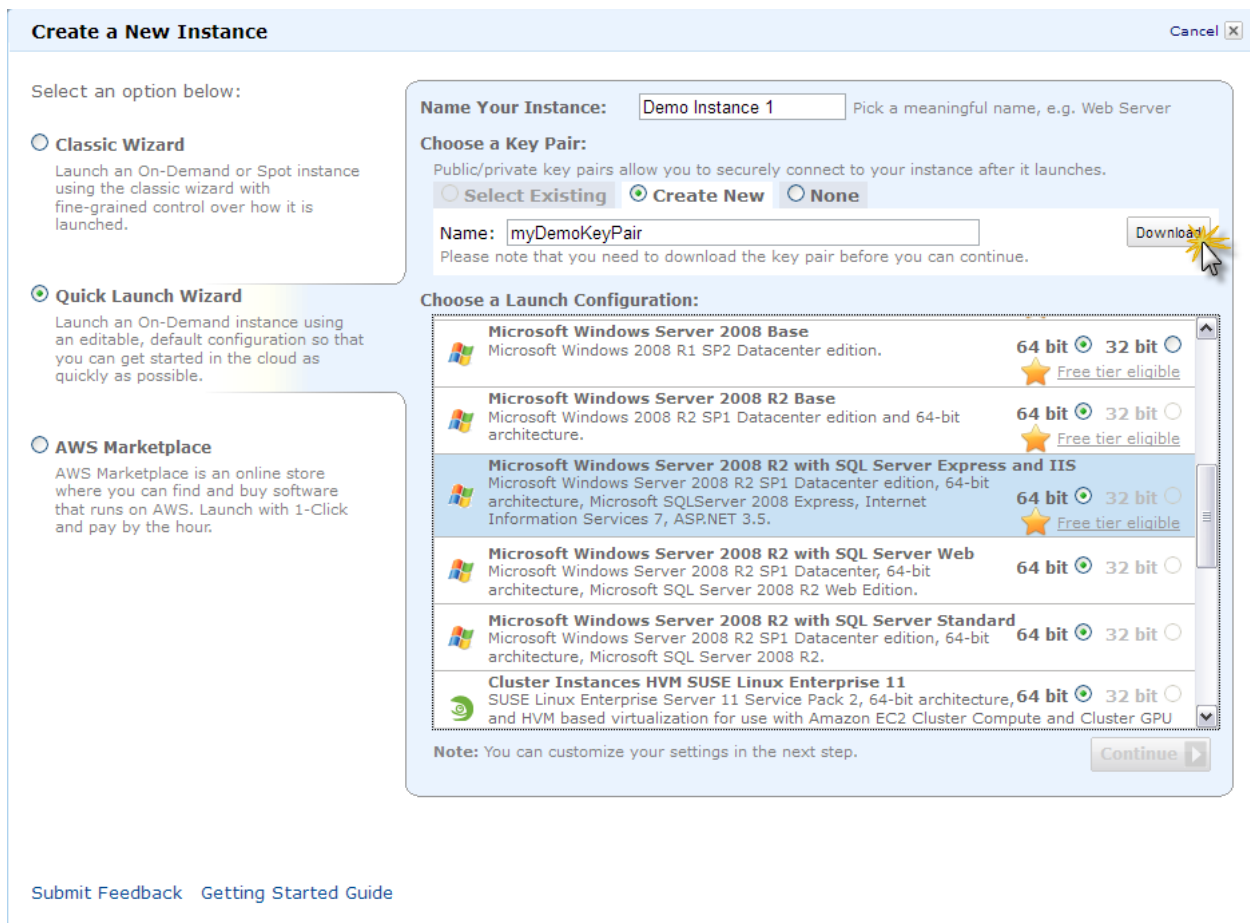


Figure 14: The Quick Launch Wizard is an easy way to get started with a new instance.

With the Quick Launch Wizard selected as shown in Figure 14, the top portion of the right-hand pane prompts you to supply a name for your instance and asks you to choose a Key Pair for security. A Key Pair is an RSA public key / private key pair generated for you by

AWS. If this is the first instance you've launched from this account, you'll need to create and download a new key pair. In Figure 14, you can see the *Select Existing* radio button is disabled because there is no key pair associated with this account yet.



When choosing a key pair, do NOT mark the *None* radio button. You'll need to supply your private key later in the server launch sequence in order to retrieve the Administrator password for your server and download the RDP shortcut file. If you do not have a private key, you'll be unable to connect to your server.

After specifying a name for your key pair, click the *Download* button as shown in Figure 14. AWS prompts you to download a .pem file with the key pair name you specified – in this example, the file name is myDemoKeyPair.pem. Save the file to a safe location on your local machine. You don't want to risk losing this file, so it's a good idea to make a backup copy at this time, too.

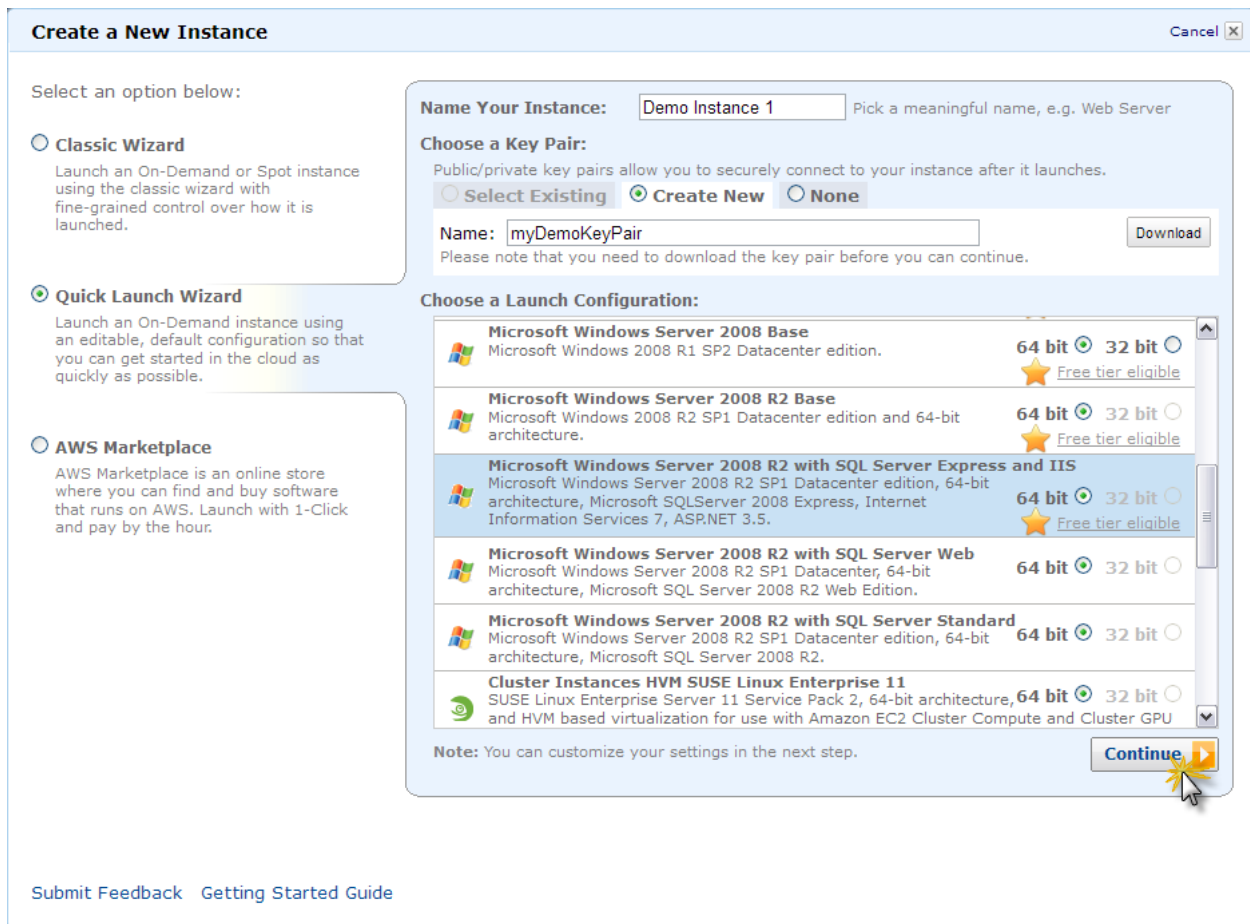


Figure 15: Select the server configuration you want to launch, then click the Continue button.

Once you've downloaded the new key pair, the *Continue* button becomes enabled in the Quick Launch Wizard. Before continuing, however, first select the server configuration you want to launch. You can scroll up and down in the Launch Configuration list to see the various types of configurations available. Figure 15 shows a portion of this list, with a particular 64-bit Windows Server configuration selected. The gold star indicates configurations that qualify for the AWS free tier usage.

Click Continue to proceed to the next step. AWS displays the details of the server configuration you chose to launch, as shown in Figure 16.

The screenshot shows the 'Create a New Instance' wizard. At the top, the title is 'Create a New Instance' with a 'Cancel' button. Below the title, the selected instance configuration is highlighted in a blue box: 'Microsoft Windows Server 2008 R2 with SQL Server Express and IIS (ami-4fcb7e26)'. The platform is 'Windows' and the architecture is 'x86_64'. The description reads: 'Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQLServer 2008 Express, Internet Information Services 7, ASP.NET 3.5.' Below this, a message says: 'Please review your settings and click **Launch** to finish or **Edit details** to make changes.'

The 'Instance Details' section includes:

- Name: Demo Instance 1
- Type: t1.micro
- Detailed Monitoring: No
- Availability Zone: No preference
- Shutdown Behaviour: Stop
- Termination Protection: No
- Launch into a VPC: No

The 'Security Details' section includes:

- Key Pair: myDemoKeyPair
- Security Group: quicklaunch-1

The 'Advanced Details' section includes:

- Kernel ID: Default
- Ramdisk ID: Default
- User Data:
- IAM Role: (with a help icon)

At the bottom left is a 'Go Back' button with a left arrow. At the bottom right are 'Edit details' and 'Launch' buttons.

Figure 16: You can view and (optionally) edit the details of the server configuration you chose to launch.

This is the final step before actually launching the instance. In this step, you can view and optionally to edit the details of your selected launch configuration. There is also a *Go Back* button you can use to back up to the previous step if you decide this is not the launch configuration you really wanted.

When you're satisfied everything is the way you want it, click the *Launch* button. The Quick Launch Wizard now shows that your instance is launching, as illustrated in Figure 17. Launching an instance normally takes only a few seconds but may take longer.



Figure 17: AWS suggests a couple of optional things you can do while your instance is launching.

As shown in Figure 17, AWS suggests a few things you can do while waiting for the launch to complete. These are completely optional; unless you want to do one of them, click on the *View your instances on the Instances page* link. This completes the Quick Launch Wizard and returns you to the EC2 dashboard. Your server instance is now running.

Exploring your new server instance

Figure 18 shows the My Resources portion of the EC2 dashboard after launching a new instance. It usually takes a few seconds, sometimes longer, to launch a new instance; if your new instance doesn't appear right away, you can click the Refresh button to update the display. The changes that resulted from launching a new instance are highlighted in Figure 18, where you can see there is now one running instance, one EBS (Elastic Block Storage) volume, one key pair, and two security groups.



Figure 18: After launching an instance, the My Resources pane shows there is one running instance along with one EBS (Elastic Block Storage) volume, one key pair, and two security groups.

The running instance is of course the one you just launched. The EBS volume was created along with the instance – more about EBS volumes in the next section. The key pair is the one you created and downloaded at the beginning of the process. The two security groups are the *default* group, which is not being used at this time, and the *quicklaunch-1* group, which was created automatically by the Quick Launch wizard. Security groups are discussed in more detail a bit later on.

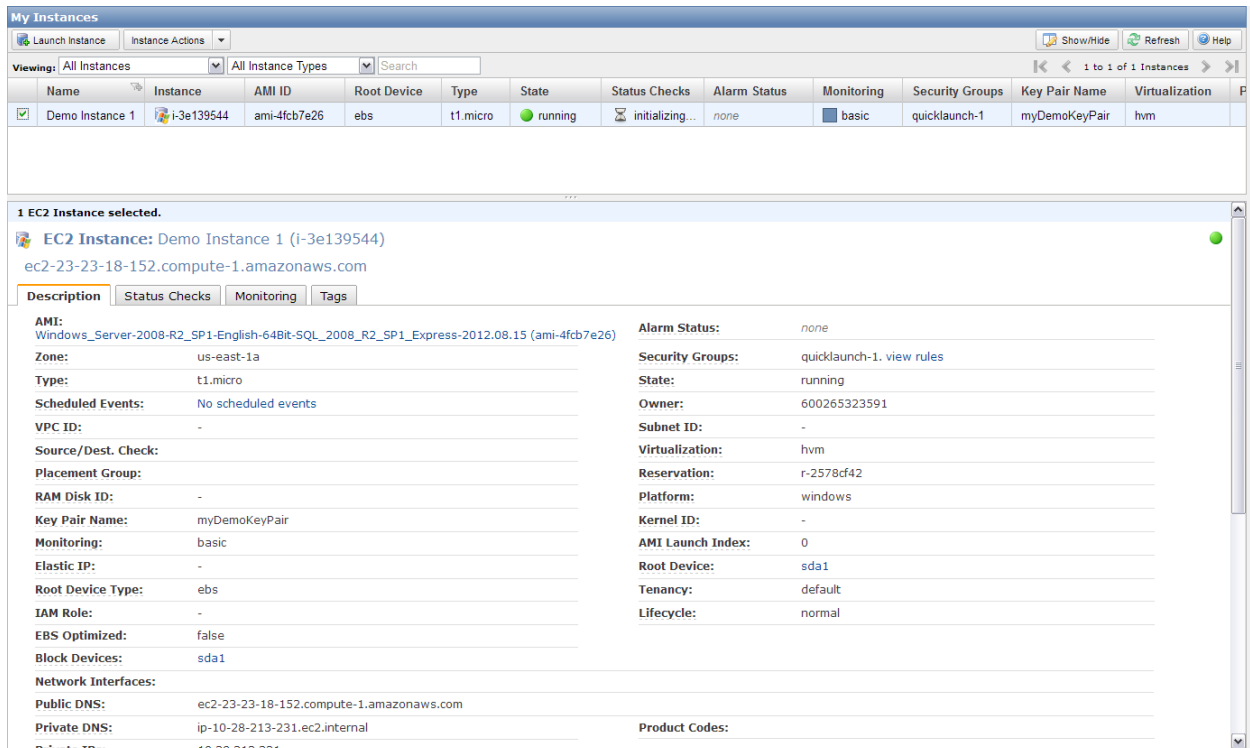


Figure 19: The My Instances page of the EC2 Console Dashboard displays a list of your instances. Details of the selected instance are shown below the list.

Clicking the *Instances* item in the EC2 dashboard navigation panel opens the My Instances page in the right-hand pane. The top of this page is a list of your instances. Figure 19 shows that *Demo Instance 1* is the only instance currently created on this account. Marking the check box to the left of its name causes the instance details to be displayed below the list.

EBS Volumes

Every EBS-backed launch configuration comes with a pre-determined amount of EBS volume storage. EBS volumes serve as hard drive space allocated to an instance; the *root volume* is the one the instance boots from. Unlike instance storage, which is ephemeral, EBS volume storage persists even when the instance is stopped.

EBS volumes exist independently of instances, and a single instance can have more than one volume attached to it. When you launch a new instance from an EBS-backed AMI, the EBS volume that's created is automatically attached as the instance's root volume. EBS volumes are identified by their Volume ID; you can also give them a name for easier reference.

Select the *Volumes* link in the EC2 Console navigation panel to view the EBS volumes associated with an account. Figure 20 shows the one created for this demo account. Reading across the top, you can see this volume's unique identifier in the Volume ID column and also that it's currently attached to the demo instance.

The screenshot shows the AWS Management Console interface for EBS Volumes. On the left is a navigation pane with 'Volumes' selected under 'ELASTIC BLOCK STORE'. The main content area shows a table of volumes with the following data:

Name	Volume ID	Capacity	Volume Type	Snapshot	Created	Zone	State	Alarm Status	Attachment Information
<input checked="" type="checkbox"/>	vol-22d0a359	30 GiB	standard	snap-3de2ea4d	2012-09-06T21:46:37	us-east-1a	in-use	none	i-3e139544 (Demo Inst

Below the table, the details for the selected volume 'vol-22d0a359' are displayed:

- Volume ID:** vol-22d0a359
- Capacity:** 30 GiB
- Created:** 2012-09-06 16:46 CDT
- Status:** in-use
- Volume Type:** standard
- Product Codes:** -
- Alarm Status:** none
- Snapshot:** snap-3de2ea4d
- Zone:** us-east-1a
- Attachment:** i-3e139544 (Demo Instance 1):/dev/sda1 (attached)
- IOPS:** -

Figure 20: Select the Volumes link to view and manage the EBS volumes associated with your account.

Security groups

Security groups are used to control access to AWS server instances. A security group is a collection of rules defining the ports that are open for inbound access and the IP addresses that are authorized to use them. An account can have more than one security group, and any given security group can be used with one or more server instances.

Figure 21 shows the two security groups currently associated with the demo account being used for this session. The default security group was automatically created when the account was set up. The quicklaunch-1 security group was automatically created by the Quick Launch Wizard used to launch the server instance, and is the one currently associated with that instance.

The screenshot displays the AWS Management Console interface for Security Groups. On the left, the navigation pane shows the 'Region' set to 'US East (N. Virginia)' and a tree view where 'NETWORK & SECURITY' is expanded to 'Security Groups'. The main content area is titled 'Security Groups' and includes buttons for 'Create Security Group' and 'Delete'. Below these are 'Viewing: EC2 Security Groups' and a search box. A table lists two security groups: 'default' and 'quicklaunch-1'. The 'quicklaunch-1' group is selected, indicated by a green checkmark. Below the table, a summary bar states '1 Security Group selected'. The details for the selected group are shown in a tabbed view with 'Details' selected. The details include: Group Name: quicklaunch-1, Group ID: sg-a2b6c3ca, Group Description: quicklaunch-1, and VPC ID: -.

	Name	VPC ID	Description
<input type="checkbox"/>	default		default group
<input checked="" type="checkbox"/>	quicklaunch-1		quicklaunch-1

1 Security Group selected

Security Group: quicklaunch-1

Details | Inbound

Group Name: quicklaunch-1

Group ID: sg-a2b6c3ca

Group Description: quicklaunch-1

VPC ID: -

Figure 21: Security groups are used to control access to a server instance.

The screenshot shows the AWS Management Console interface for Security Groups. At the top, there are buttons for 'Create Security Group', 'Delete', 'Show/Hide', 'Refresh', and 'Help'. Below this, a navigation bar shows 'Viewing: EC2 Security Groups' and a search field. A table lists two security groups: 'default' and 'quicklaunch-1'. The 'quicklaunch-1' group is selected. Below the table, a section titled '1 Security Group selected' shows the details for 'quicklaunch-1'. The 'Inbound' tab is selected, displaying a table of rules:

TCP Port (Service)	Source	Action
80 (HTTP)	0.0.0.0/0	Delete
1433 (MS SQL)	0.0.0.0/0	Delete
3389 (RDP)	0.0.0.0/0	Delete

On the left, a 'Create a new rule' dialog is open, showing a 'Custom TCP rule' configuration. The 'Port range' is set to '0.0.0.0/0' and the 'Source' is also set to '0.0.0.0/0'. There is an 'Add Rule' button and an 'Apply Rule Changes' button.

Figure 22: The Inbound tab of the security groups page shows the rules associated with the selected group.

Figure 22 shows the quicklaunch-1 security group selected at the top and the Inbound tab selected in the lower portion of the window. It's here that you can view, edit, add, and delete the rules associated with this security group. Because we started with a launch configuration that included IIS and MS SQL Server, the Quick Launch Wizard automatically set up TCP rules with ports 80 and 1433 open for access. Port 3389 is also open by default so you can access your server instance via RDP.

Note that the Source field is initially set to 0.0.0.0/0 for all three rules, meaning the ports are accessible from any inbound IP address. If you're planning to use your server to host a publicly accessible web site, then this is what you want for port 80.⁵ It almost certainly is not what you want for your SQL Server, and it may or may not be what you want for your RDP access.

For security reasons, Amazon recommends restricting RDP access to a single inbound IP address – otherwise it's "open to the world" and protected only by your Windows Administrator account password. However, if you want to be able to RDP into your server regardless of where you might be geographically at any given time, say for support purposes, then you probably don't want to restrict RDP access to a single inbound IP address.

You can add several types of rules to a security group. The *Create a new rule* drop down list shows the available choices, as shown in Figure 23. When you're working with TCP or UDP rules you can also specify a range of ports to fine-tune the rule.

⁵ Your server instance doesn't yet have a publicly accessible IP address. More on this later.

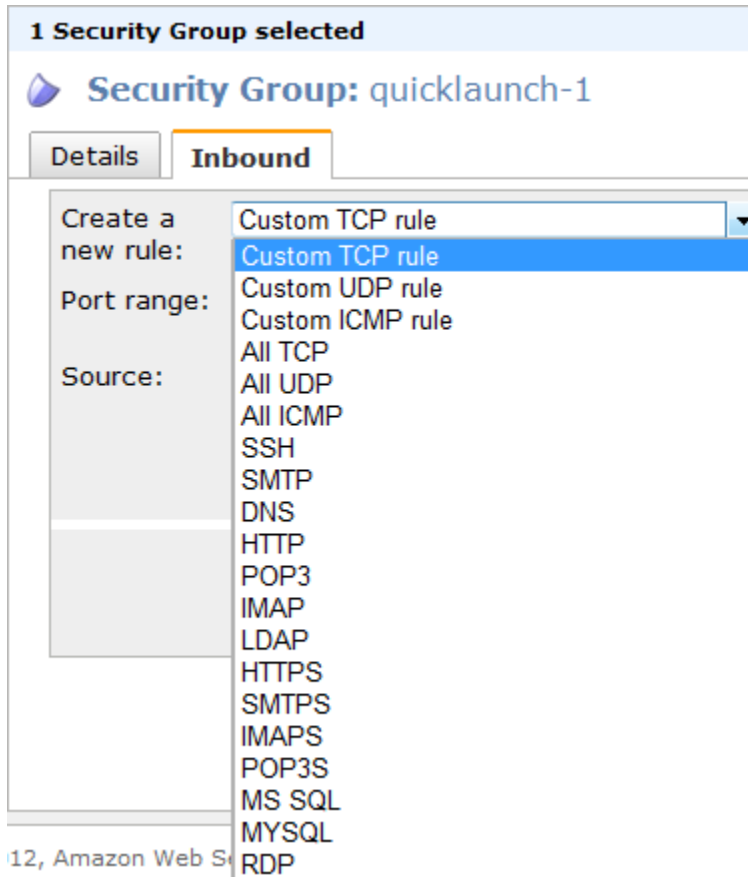


Figure 23: You can create several different types of rules to control access to your server instance.

The Source field for a security group rule can hold either an IP address or an EC2 security group ID. The former can be either a single IP address or a range of IP addresses. If you specify a security group, the format of the value you enter depends on whether it's the current security group, another security group in your account, or a security group from another account. The Amazon EC2 Microsoft Windows Guide referenced earlier has a chapter on controlling access with security groups and credentials; this is your best source for further information.

Connecting to your server

Although your server instance is now up and running, you can't really do anything with it until you connect to it. Connections to AWS servers running Microsoft Windows Server software are made via RDP. Once connected, you have RDP access to the server's Windows desktop, and from there it's just like working with any other Windows server.

The first time you connect to a server instance, you need to do so from the EC2 Management Console. AWS walks you through a series of steps to establish your

credentials, then retrieves the Administrator password for you and enables you to download the RDP shortcut file for that instance. From then on you can connect either from the Management Console or by using the RDP shortcut, although the latter is much more convenient.

To begin the process of connecting for the first time, select the Instances item in the dashboard's navigation panel. Then, in the list of instances at the top of the My Instances display, mark the check box on the line for the server to which you want to connect.

AWS provides an *Instance Management* menu for the purpose of managing your server instances. It's a context (pop-up) menu, which you can get to by clicking the Instance Actions drop-down button near the top or by right-clicking on the selected instance in the list. The latter seems less than intuitive to me because, in general, we're not used to right-clicking on things in a browser window unless we want the browser's own context menu, but it works.

Figure 24 shows the Instance Management context menu for the demo instance. The various menu items are enable or disabled depending on the state of the server and other factors. Note the *Connect* item selected at the top of the menu. Click this item to initiate the connection process.

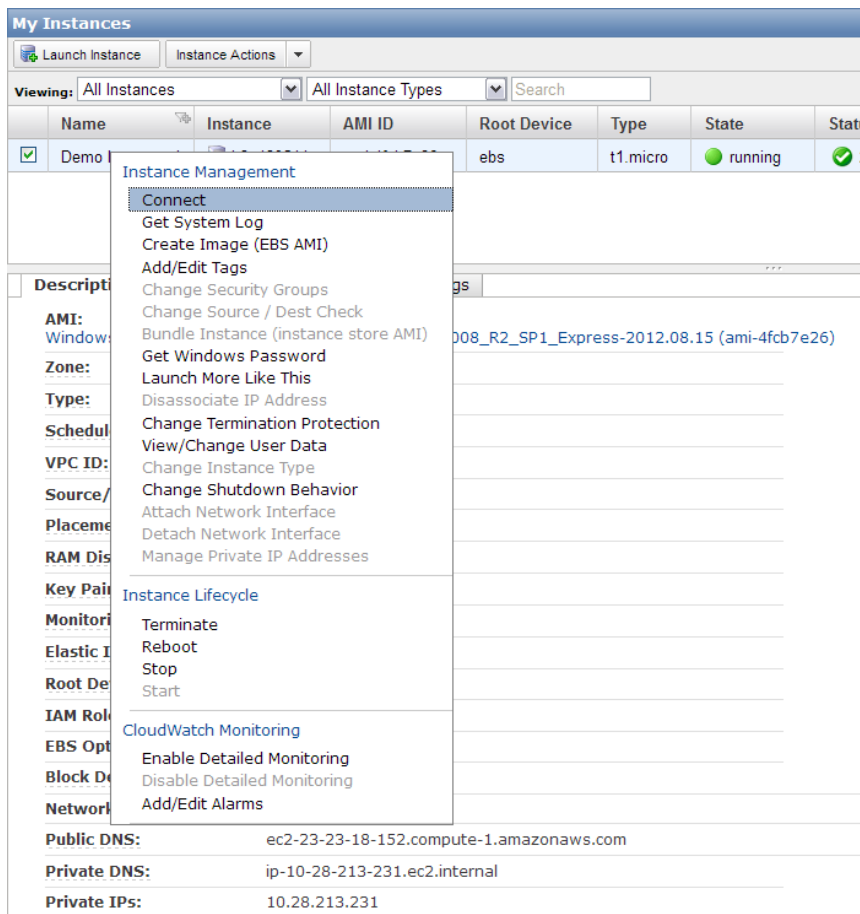


Figure 24: AWS provides an Instance Management context menu to manage your server instances.



I ran into a minor annoyance with the context menu using Firefox. In response to the right-click, Firefox also brings up its own context menu and superimposes it on top of the EC2 Instance Management menu. This obscures some of the AWS menu items, as shown in Figure 25. Pressing the Escape key gets rid of the Firefox menu and leaves the AWS Instance Management menu visible and accessible, so there's no real loss of functionality, but it's an annoying quirk. I did not observe this behavior when using Microsoft Internet Explorer or Google Chrome.

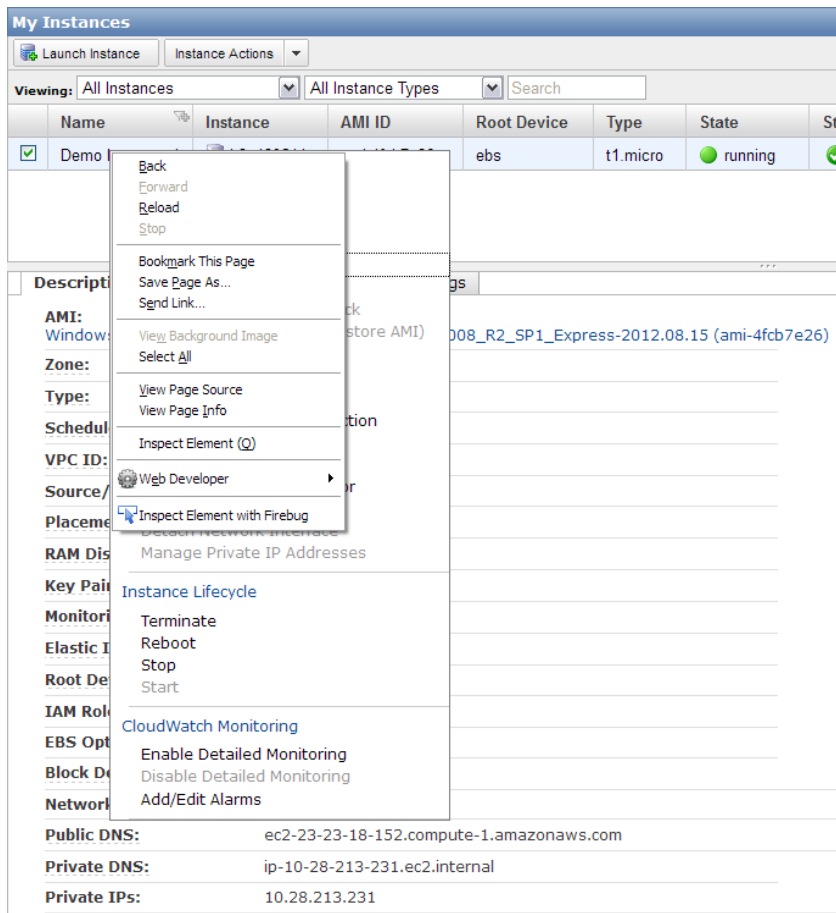


Figure 25: Firefox brings up its own browser context menu and superimposes it on top of the AWS Instance Management menu. Pressing the Escape key gets rid of it.

When you connect to your server instance from the EC2 dashboard Instance Management context menu, AWS brings up the screen shown in Figure 26. Click on the Retrieve Password link to go to the next step.

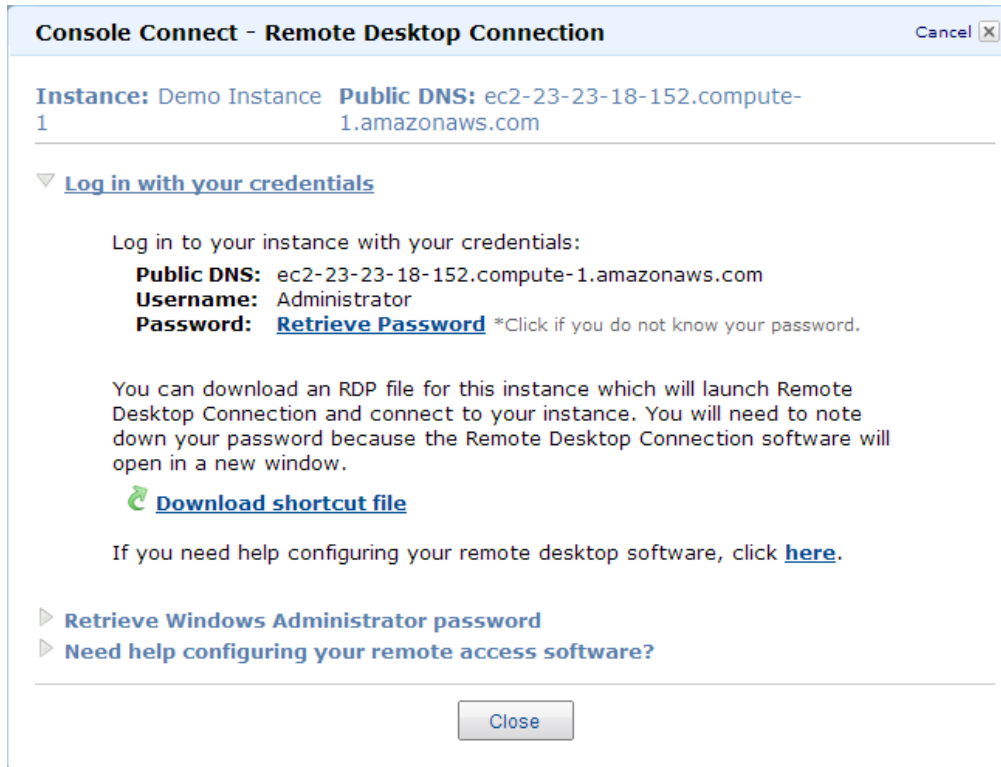


Figure 26: Click the Retrieve Password link to get the Administrator password for your new server instance.

In the next step you are prompted to supply the private key from the key pair you created earlier. Click the Browse button and navigate on your local file system to locate and select the .pem file you downloaded earlier. Selecting this file places the contents of your private key in the appropriate field on the screen. When that's been done, click the Decrypt Password button to proceed to the next step, as illustrated in Figure 27.

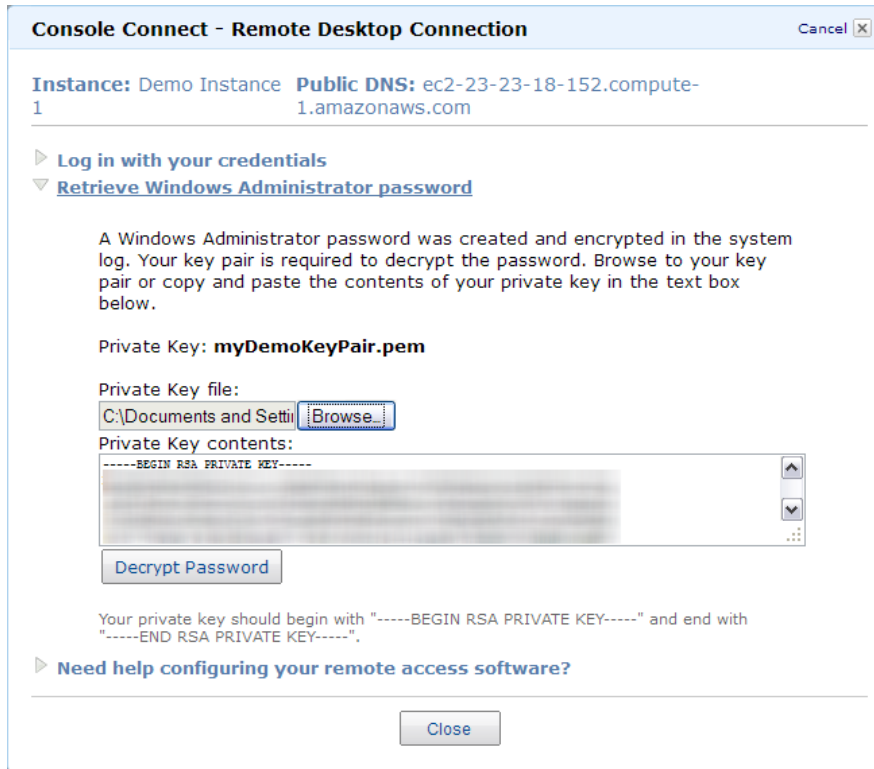


Figure 27: Use the Browse button to locate and select the .pem file you downloaded earlier. Then click the Decrypt Password button.

AWS generates a strong password for your server instance and creates an RDP shortcut file. Click the *Download shortcut file* link as shown in Figure 28, and save the RDP file to a convenient location on your local file system. In the future, you can use this shortcut to connect to your server instance without having to repeat these steps.

As part of this same step, AWS displays the Administrator password for your server instance (see Figure 28). You'll need this password to log in to your server, so be sure to write it down and record it in a secure location. However, if you forget or lose track of the password, you can repeat these steps to retrieve it again.

AWS recommends you change the Administrator password right away, but this is optional. AWS also recommends you create another user account with Administrator privileges, which you can do in the conventional manner as soon as you complete the log in process, in case you have problems logging in with the Administrator account.

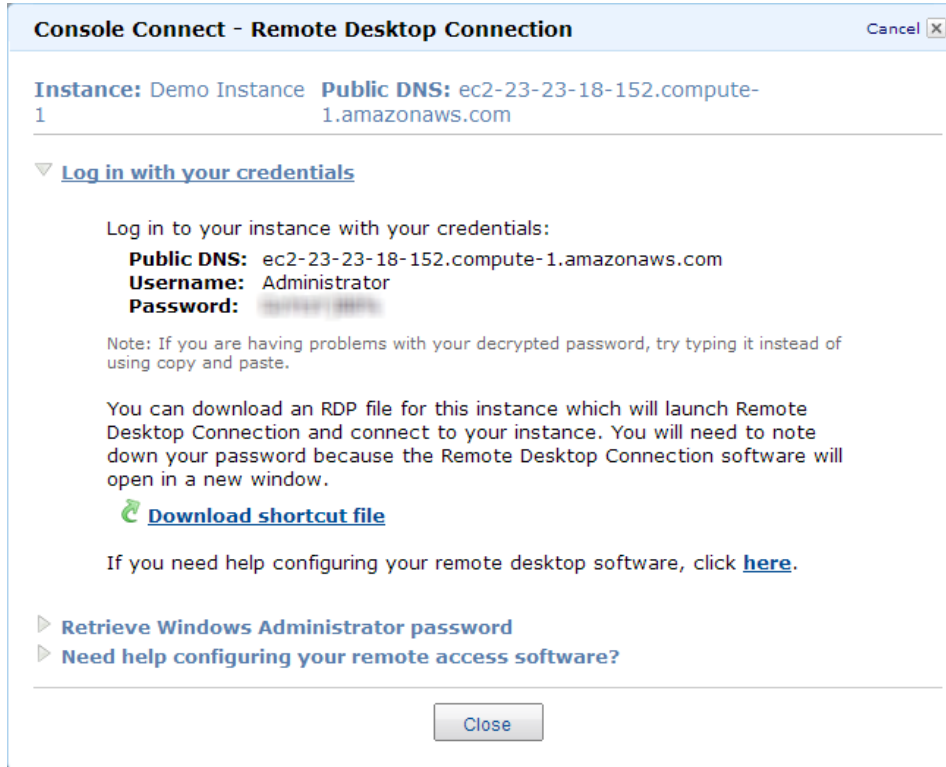


Figure 28: AWS displays the Administrator password generated for your server instance and provides a link to download the RDP shortcut file.

Once you have the RDP shortcut file, use it to connect to your server instance. Log in as Administrator with the password assigned by AWS. By default, the RDP file enables sharing of printers and the clipboard. You can edit this file just as you would edit any other RDP file if you want to share other resources, such as your local drive(s).

Figure 29 shows the Windows Desktop in an RDP session on the local machine after completing the connection to the server. I opened the System Properties window manually.

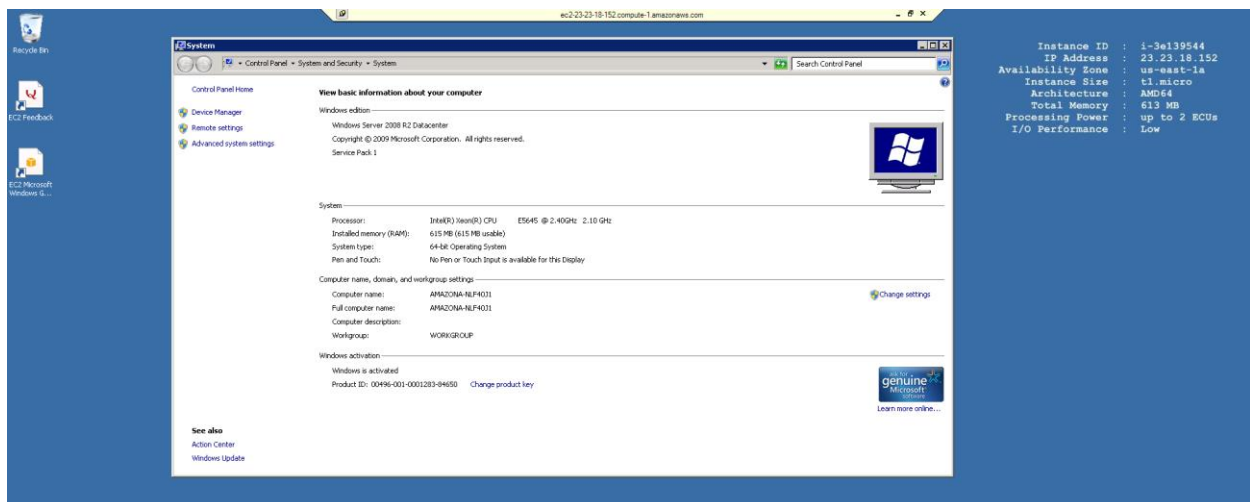


Figure 29: The Windows Desktop of an AWS server instance, displayed in an RDP terminal session on the local machine. I opened the Computer Properties window manually after logging in.

Congratulations – you’re up and running with EC2!

Working with your server instance

At this point you have launched a new server instance, established security, retrieved the Administrator password, and established a connection using RDP. Your primary tools for working with your server from here on are the EC2 Management Console, which you’ll use for the management functions and services provided by AWS, and the RDP terminal session you’ll use for working with the server itself.

The EC2 Management Console

You have already seen the EC2 Management Console in many of the previous steps, so it should be somewhat familiar to you by now. There are several parts to the Management Console. This section is a brief review of some that you have already seen and an introduction to some you haven’t used before.

The Console Dashboard

The EC2 Console Dashboard, or just the “dashboard” for short, is basically the home position of the Management Console. You get to it by clicking the *EC2 Dashboard* link at the top of the navigation panel on the left side of the display (see Figure 13). The items that are most likely going to be of interest to you on an ongoing basis are the current status of AWS services in the region you are using, which is displayed in the Service Health portion of the dashboard, and the summary of the resources you are currently using, which is displayed in the My Resources portion.

My Instances

You’ve already seen the My Instances panel, as shown back in Figure 19, to explore and connect to your server instance. This is probably the page you will use most often when working with your server. It’s from here that you can view the details of your server instance, start and stop it, and perform other management functions.

If you get to the point where you’re managing several AWS instance, the interface provides filters you can set to view only those you’re currently interested in. Choices include running instances, stopped instances, and terminated instances, among others. The default setting for these filters is All Instances and All Instance Types, as shown in Figure 30.

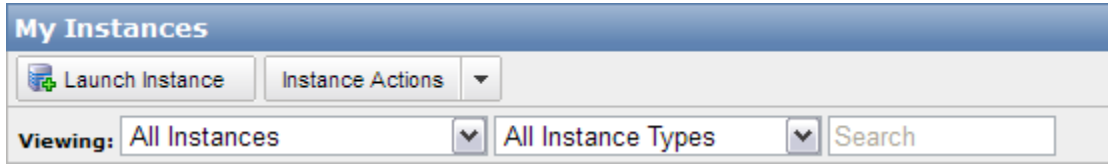


Figure 30: If you're working with a long list of server instances you can set filters to view only those of interest.

Stopping a running instance

Depending on your needs and the role your server is tasked to do, you may not want to leave it running 24/7. Even if it is tasked to run 24/7, you may need to temporarily stop it to apply updates or to perform management functions. Note that stopping an instance is not the same as rebooting it; rebooting is discussed a bit later on.

To stop a running instance, select it in the My Instances panel, bring up the Instance Management menu, and choose Stop as shown in Figure 31.

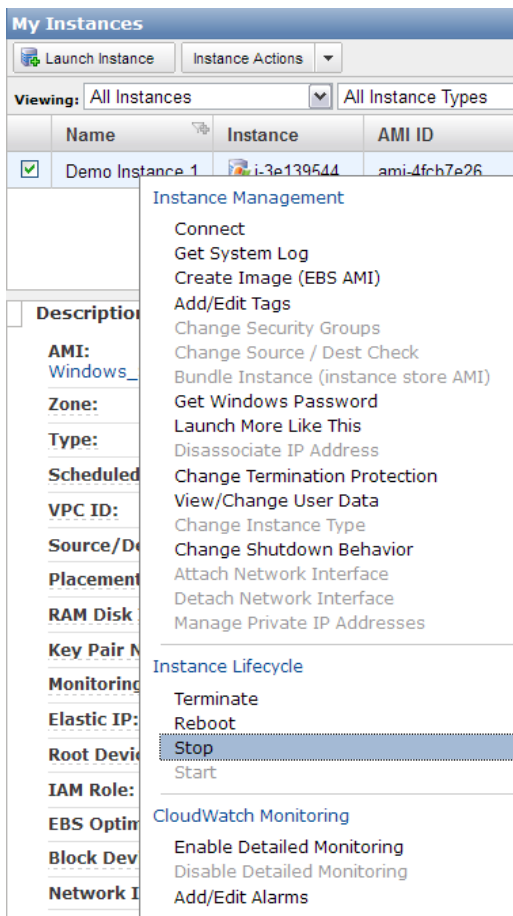


Figure 31: You can stop a running instance from the Instance Management menu.

Before actually doing so, AWS asks you to confirm that you want to stop the instance. When you stop an instance, it's like shutting down Windows – whatever is running stops running and whatever is in RAM memory is lost. It's a good idea to put your instance in a stable state before stopping it by ensuring that all applications have been terminated normally and all important files have been saved and closed. Note that stopping an instance does not affect its EBS volume storage, which persists and is available again when the instance is restarted.

When you're ready to proceed, click the *Yes, Stop* button in the confirmation dialog, as illustrated in Figure 32.

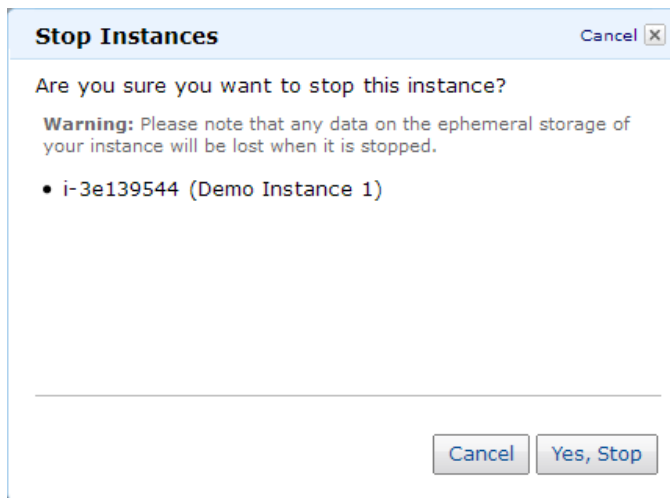
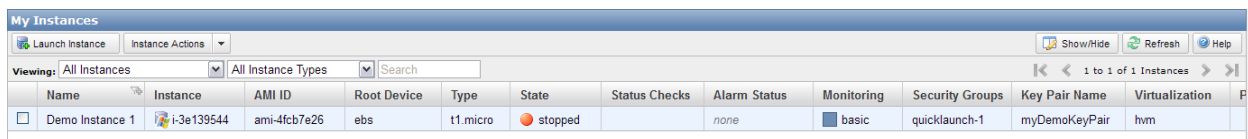


Figure 32: Click the *Yes, Stop* button to confirm you want to stop a running instance.

It normally takes only a few seconds for the instance to be stopped. Until that happens, the State column in the My Instances display will show a transition status. When the transition is complete, the State column shows that the instance is stopped, as illustrated in Figure 33.



Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Groups	Key Pair Name	Virtualization	
<input type="checkbox"/>	Demo Instance 1	i-3e139544	ami-4fb7e26	ebs	t1.micro	stopped		none	basic	quicklaunch-1	myDemoKeyPair	hvm

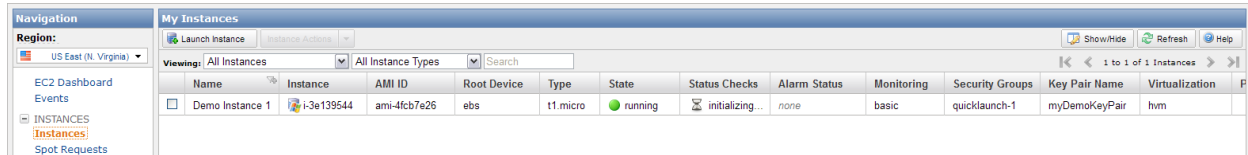
Figure 33: The *State* column shows this instance has been stopped.

Keep in mind that you cannot connect to a stopped instance. You'll need to start it again before you can connect to it.

Starting a stopped instance

It's easy enough to restart a stopped instance. Simply select the desired instance in the My Instances list, bring up the Instance Management menu, and click Start. The State column

shows a transition status while the server is being started. When it's fully running again, the State column shows the running status, as illustrated in Figure 34.



The screenshot shows the AWS Management Console 'My Instances' page. The region is set to 'US East (N. Virginia)'. A table lists one instance: 'Demo Instance 1' with Instance ID 'i-3e139544', AMI ID 'ami-4fc7e26', Root Device 'ebs', Type 't1.micro', and State 'running'. The Status Checks column shows 'initializing...' with a green dot, and Alarm Status is 'none'. Other columns include Monitoring (basic), Security Groups (quicklaunch-1), Key Pair Name (myDemoKeyPair), and Virtualization (hvm).

Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Groups	Key Pair Name	Virtualization
Demo Instance 1	i-3e139544	ami-4fc7e26	ebs	t1.micro	running	initializing...	none	basic	quicklaunch-1	myDemoKeyPair	hvm

Figure 34: The *State* column shows this instance is running.



You need to be aware of one important side effect to stopping and starting an instance: unless you're using a static IP address (explained in the next section), your server will be assigned a new IP address when you restart it and you'll need to download a new RDP shortcut file in order to reconnect to it.

AWS assigns each instance a private host name and a public host name. You can see them on the Private DNS and Public DNS lines of the EC2 Management Console when an instance is selected. The RDP shortcut for an instance is based on the IP address assigned to its public host name. When you restart a stopped instance, AWS assigns a different IP address and the instance therefore has a different public host name. This means the RDP shortcut file you used to connect to the instance before you stopped it cannot be used to connect to the server after it is restarted, because that shortcut was based on the previous IP address.

To connect to the server after restarting it from a stopped state, you need to step through the EC2 Console connection process again, supplying your credentials and downloading a new RDP shortcut file. The password for the Administrator account is normally not affected by doing this, so it remains the same as before.⁶

This problem can be avoided by requesting a static public IP address from AWS and associating it with your server instance. Amazon refers to these as Elastic IP Addresses, or EIPs. Elastic IP addresses are covered in the next section.

Obtaining a static IP address

If you want your server to be accessible to the outside world, for example as a Web server, you'll need to give it a static public IP address. Even if you don't intend for your server to be accessible by the general public, giving it a static IP address gives you the advantage of not having to download a new RDP shortcut file every time you stop and restart the instance.

You can request AWS to allocate one or more public static IP addresses to your account. Amazon calls these Elastic IP Addresses, or EIPs. Obtaining an EIP is easy. Start by selecting

⁶ This can be affected by changes made to the instance's configuration via the EC2 Configuration Service Utility. See the section on Creating Your Own AMI.

the *Elastic IPs* link in the EC2 Management Console navigation pane, then click on the *Allocate New Address* button as shown in Figure 35.

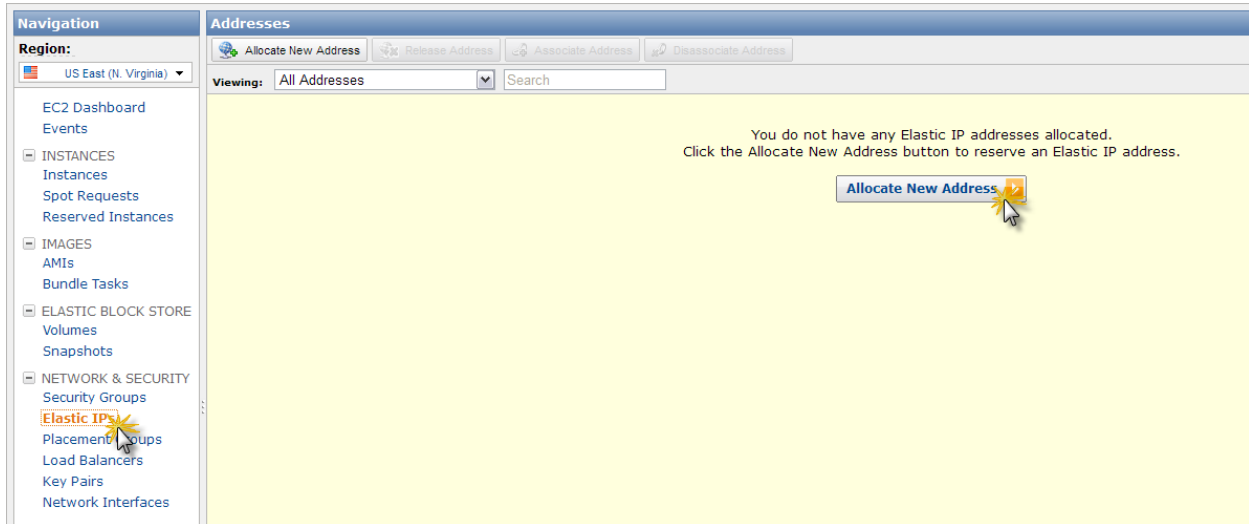


Figure 35: To obtain a public static IP address for your instance, start by selecting the *Elastic IPs* link on the EC2 Management Console navigation pane, then click *Allocate New Address*.

Click the *Yes, Allocate* button in the confirmation dialog.

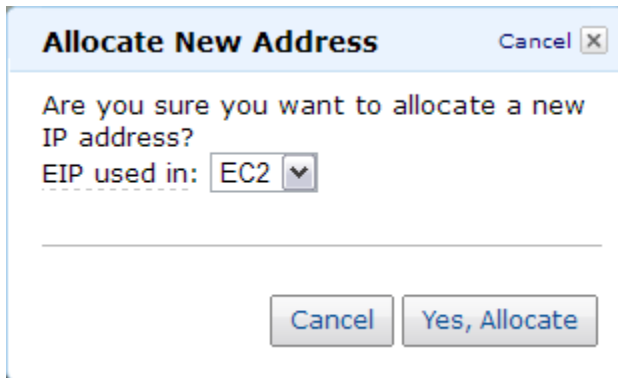


Figure 36 Click the *Yes, Allocate* button to confirm you want to allocate a new IP address.

Amazon allocates a public static IP address to your account. At this point the IP address belongs to your account but is not yet associated with any of your instances. Click the *Associate Address* button to associate the address with the desired instance, as shown in Figure 37.



When you associate the new public static IP address with your server instance, you'll want to download the new RDP shortcut file that's based on the new IP address. To do this, connect to your instance from the EC2 Management Console,

supply your credentials, and download the new shortcut file. Because the IP address is now static, you won't have to go through this process again unless you change the instance's IP address in the future.

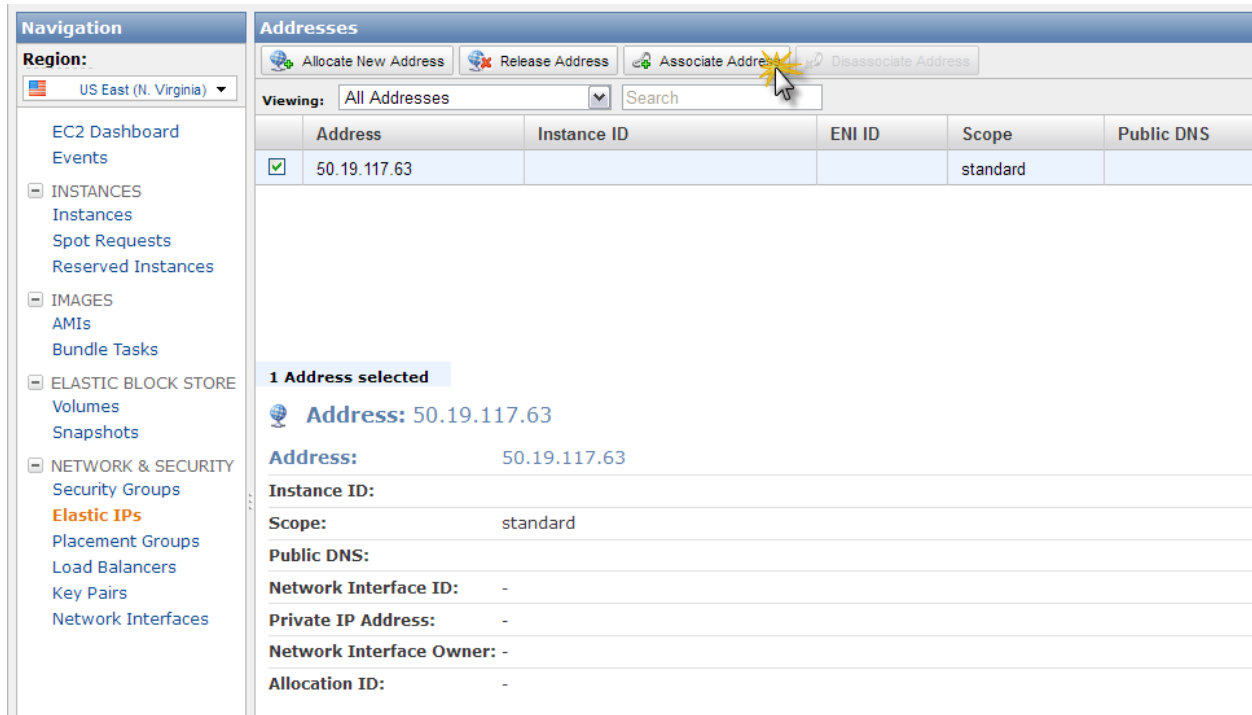


Figure 37: Click the *Associate Address* button to associate a public static IP address with an instance.

Select the desired instance from the drop-down list, then click *Yes, Associate*.

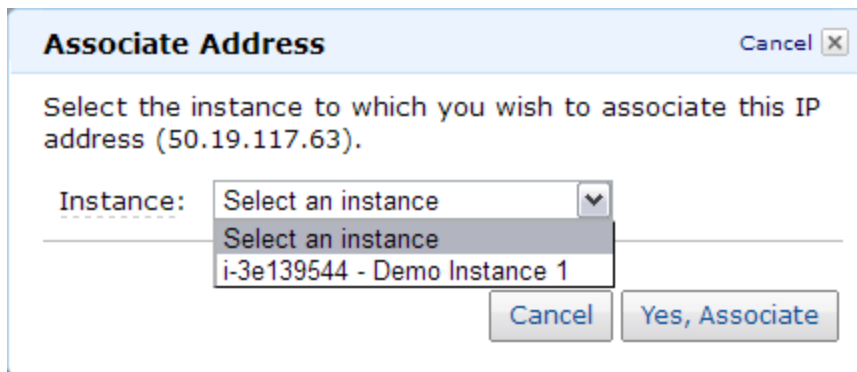


Figure 38: Select the desired instance from the drop-down list, then click the *Yes, Associate* button.

The Management Console shows that the IP address is now associated with the instance you selected.

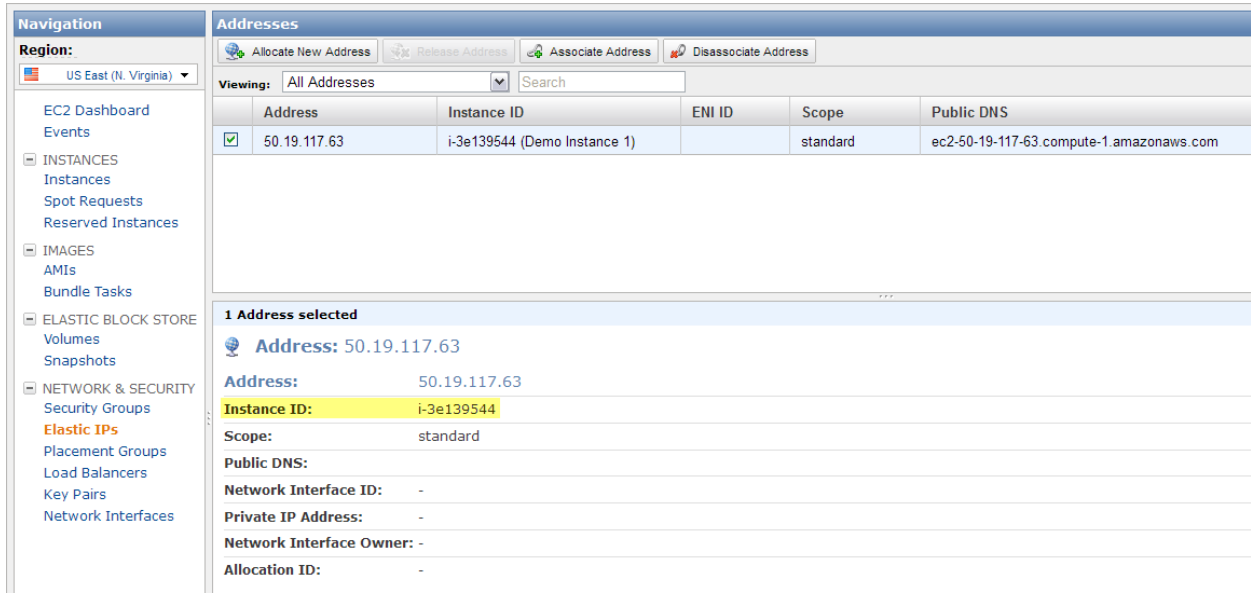


Figure 39: The Management Console now shows that the IP address is associated with a particular instance ID (highlighting added).

Your server is now publicly accessible via its EIP address, subject to the rules in the associated security group. If your server is running IIS and port 80 is open to all inbound traffic, you can enter the EIP address in your browser and the server should respond with the default web page, as shown in Figure 40.



Figure 40: After you associate an elastic IP address with your server instance, your server is accessible from the outside world – subject to the rules in its security group.

You can disassociate an elastic IP address from an instance and later re-associate with the same or with another instance. There is no charge for an EIP as long as it is associated with a running instance; otherwise there is a small hourly fee. Either way, the EIP remains yours until you release it from your account.



Note that stopping a running instance automatically disassociates its elastic IP address. However, the EIP is still assigned to your account so you can re-associate it when you restart the instance.

Rebooting your server

If you have an RDP terminal session open, you can reboot your server from Windows in the conventional manner. You can also reboot an instance by using the Reboot option on the Instance Management menu in the EC2 Management Console.

I have to admit I was somewhat apprehensive the first time I needed to reboot my AWS instance. Although most of me knew better, a small part of me was nonetheless afraid my tediously configured server—which by that time was running a client’s live e-commerce website—would vanish into the cloud along with my app and all of the user’s data, never to be seen again.

As you can predict, I need not have worried: rebooting an AWS instance works just like rebooting a physical server. Of course, if you reboot from an RDP terminal session the session gets closed, but after giving the server a couple of minutes to restart you can simply re-connect.

Unlike stopping an instance, rebooting it does not disassociate its elastic IP address.

Backing up your server

Short of attaching a physical backup device, you can back up your AWS server the same way you would back up a physical server. In addition, AWS provides some features you can use to backup your server instances and volumes in ways you can’t do with a physical server.

Using a feature called EBS volume snapshots, you can create moment-in-time backups of a server instance’s hard drive(s). In the event of a failure, you can fire up a new instance, detach its default volume, and attach the most recent snapshot volume to get up and running again.

Another way to back up your server is to create your own Amazon Machine Image (AMI). You’ll find information on creating AMIs a bit later in this paper.

Another thing you can do to reduce the risk of loss associated with a hard failure of your server instance—not to mention a failure or prolonged outage of the AWS infrastructure itself—is to store your snapshots and/or AMIs in other availability zones within the same

region or even in a different AWS region. Availability zones are isolated from one another and AWS regions are in geographically dispersed locations, so having backup resources in multiple places reduces the chance that they'll all be affected by the same event.⁷

Terminating your server instance

Terminating an instance permanently removes it from your AWS account. Once the termination process is initiated it cannot be stopped, and once an instance has been terminated it cannot be recovered. Termination is therefore something you want to treat carefully, doing it only when you're certain you're finished with an instance and won't need it or its resources again.

To terminate an instance, select the instance in the My Instances list in the EC2 Console and choose Terminate on the Instance Management menu. AWS prompts you to confirm you really want to do this, as shown in Figure 41.

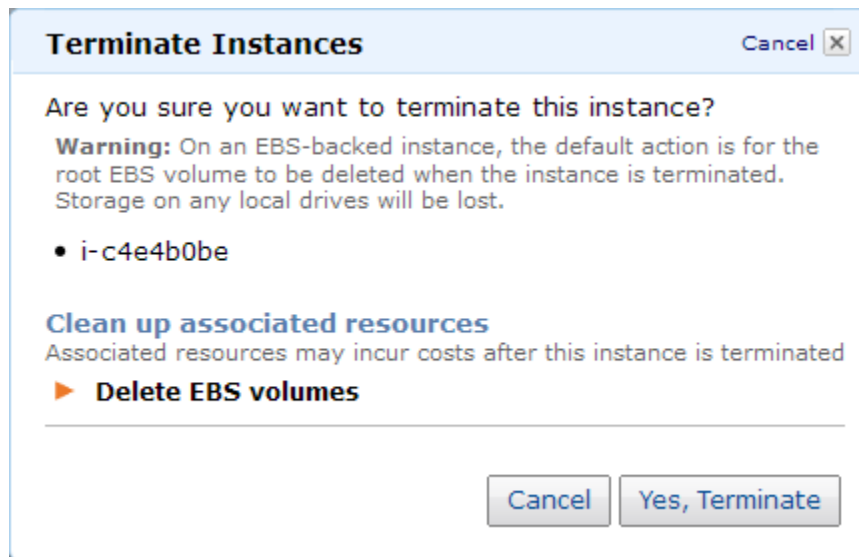


Figure 41: Termination is an irreversible action, so AWS gives you a chance to cancel before proceeding.



Before proceeding, pay attention to the warnings in this dialog. By default, terminating an instance also deletes its EBS root volume. When an EBS volume is deleted, the data on it is lost.

When you're ready to proceed, click the *Yes, Terminate* button. AWS initiates the termination process, which can take a few minutes while the instance and its resources are removed. When termination is complete, the My Instances list shows the instance in terminated status, as shown in Figure 42.

⁷ The massive June, 2012 outage caused by severe storms that affected Amazon's Northern Virginia data center are a good case in point.

Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Groups	Key Pair Name	Virtualization
<input type="checkbox"/> Demo Instance 1	i-3e139544	ami-4fcb7e26	ebs	t1.micro	running	2/2 checks passed	none	basic	quicklaunch-1	myDemoKeyPair	hvm
<input type="checkbox"/> empty	i-c4e4b0be	ami-21853548	ebs	t1.micro	terminated		none	basic	quicklaunch-1	myDemoKeyPair	hvm

Figure 42: The State column shows that the second instance in this list has been terminated.

I'm not sure how long a terminated instance continues to show up in the My Instances list in the EC2 Management Console, because in my experience it soon disappears entirely. The associated EBS volume that was removed when the instance was terminated also gets removed from the Volumes list.

Termination protection

Unlike merely stopping a server instance, an action that can be reversed by simply restating it, terminating an instance is a permanent action that can't be undone. AWS offers a feature called termination protection to protect your instances from inadvertent termination.

You can enable Termination Protection when an instance is launched by marking the appropriate check box during the launch sequence. To enable termination protection on an instance that's already running, select the instance in the My Instances list and choose Change Termination Protection from the Instance Management menu. This acts as a toggle: if termination protection is disabled you can enable it, and vice versa.

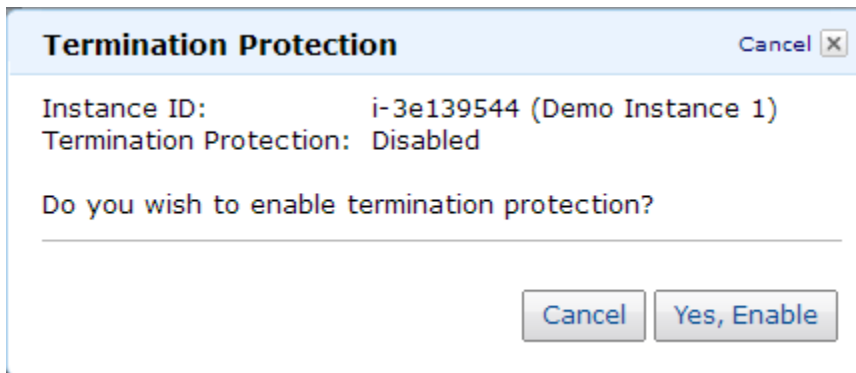


Figure 43: Termination protection is disabled by default but can be easily enabled.

The current status of termination protection for an instance is displayed along with the other details when the instance is selected in the My Instances list in the EC2 Management Console.

Leveraging your setup

By the time you get your AWS server instance launched, configured, fine-tuned, with applications installed and everything up running the way you want it, you have probably

invested a significant amount of time. Wouldn't it be nice if you could leverage that investment by packaging it up and using it to create future instances of the same or similar type? Fortunately you can, by creating volume snapshots and even your own AMIs.

Snapshots

Snapshots are a moment-in-time image of a volume. They are useful for many reasons. For example, you can use periodic snapshots to create a series of back-ups of your instance's hard drive(s) over time. You can also use a snapshot as a way of replicating a server by detaching a new instance's default volume and attaching a snapshot in its place.

Before creating a snapshot, it's advisable to put your instance in a stable state by terminating any running instances, saving and closing any important files, etc.

There are a couple of ways to create a snapshot. One is to start by selecting the *Snapshots* link in the console's navigation pane and then click the Create Snapshot button, as shown in Figure 44.

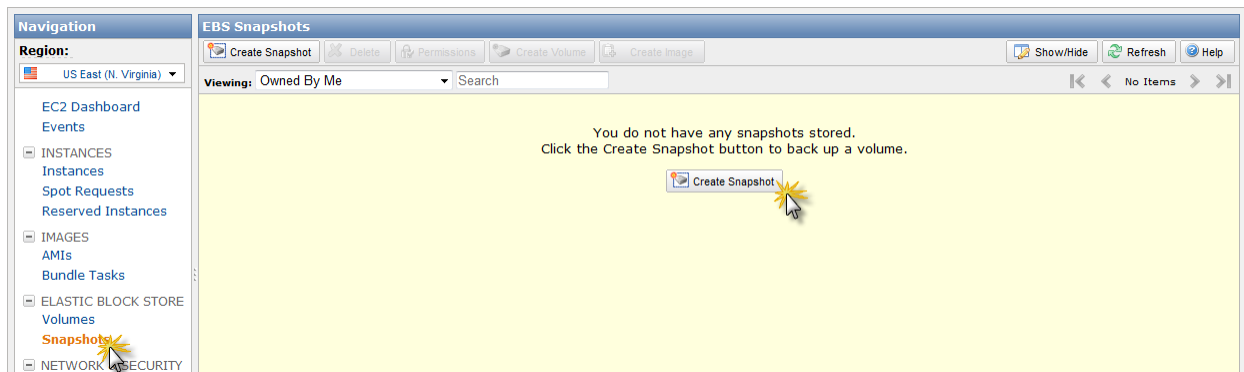


Figure 44: You can create and manage EBS volume snapshots from the EC2 Management Console.

In the Create Snapshot dialog, select the desired volume from the drop-down list and click the *Create* button. You can also give your snapshot a name and a description.

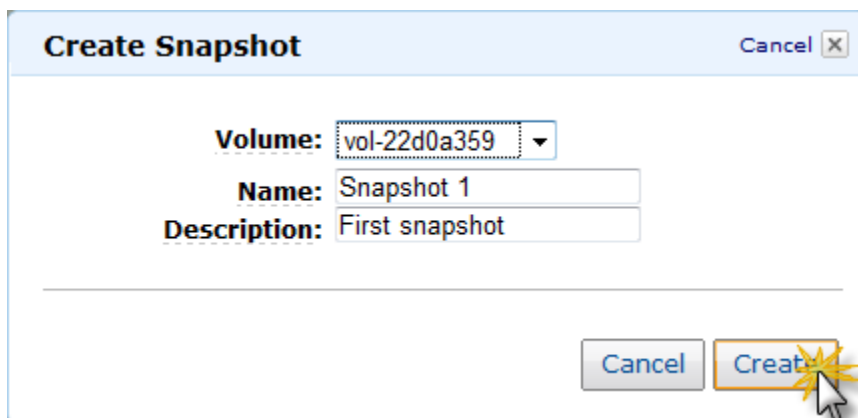


Figure 45: Select the desired volume and give the snapshot a name and brief description.

AWS creates the snapshot and adds it to the list in the Snapshots panel. Snapshots are assigned a Snapshot ID, as shown in Figure 46. It typically takes a little while to create the snapshot. The Status column shows *completed* when the snapshot is done.

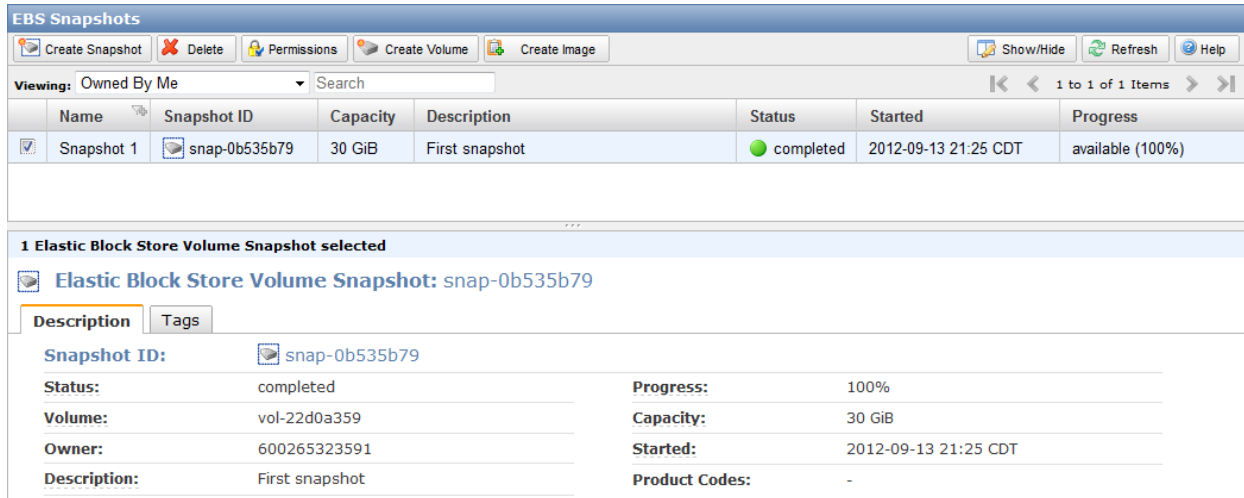


Figure 46: Snapshots are assigned a unique Snapshot ID. The list also displays the snapshot’s name and description.

Another way and perhaps more direct way to create a snapshot is to start from the EC2 Console’s Volumes pane. Select the desired volume in the Volumes list, right-click, and select Create Snapshot from the context menu as shown in Figure 47. You’ll still want to come back to the Snapshots page to view and manage your snapshots, though.

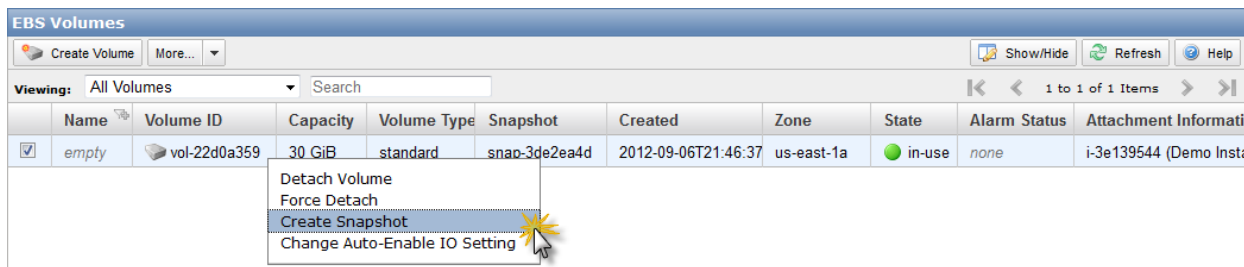


Figure 47: You can also create a snapshot using the Instance Management menu on the EBS Volumes page.

Creating your own AMI

One of the coolest things about AWS is that you can create your own Amazon Machine Image (AMI) from an instance you have already launched and configured. The AMI gives you the ability to launch new server instances with the same configuration as the original.

The basic procedure for creating an AMI is easy and straightforward, and is covered in this section. There is quite a bit to know beyond the basics, though; your best source for full information is the chapter on Using Windows AMIs in the EC2 Microsoft Window Guide

Before creating your own Windows AMI from a running instance, you need to use the Amazon EC2 Configuration Service utility to apply the settings that establish the baseline behavior of the instances that will launched from your AMI. In particular, unless you are supplying your own Administrator password, you need to mark the check boxes that tell EC2 to generate a new random password for each instance the first time it's launched. If you don't do this, you will be unable to connect to any instance launched from your AMI.

The EC2 Configuration Service utility is a file named EC2ConfServiceSettings.exe located in the C:\Program Files\Amazon\EC2ConfigSevice folder on your server. Run it from an RDP connection to a running instance of the server you're configuring to become an AMI. For complete information on the settings you can control with this utility, refer to the *Creating Your Own Windows AMI* section in the EC2 Microsoft Window Guide.

After you've prepared the instance with the EC2 configuration utility, you're ready to create the AMI. Begin by selecting the instance in the My Instances list of the EC2 Management Console. Amazon recommends the instance be stopped before creating the AMI in order to insure the server is in a stable state, although this is not a strict requirement. When you're ready to proceed, choose *Create Image (EBS AMI)* from the Instance Management menu, as shown in Figure 48.

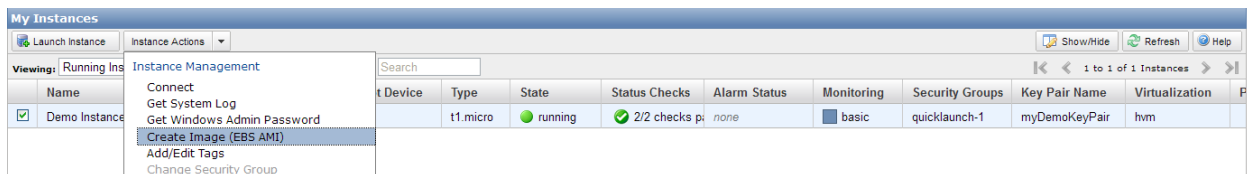


Figure 48: You can create an AMI of a running instance from the Instance Management menu.

AWS walks you through a series of steps to create an AMI of the selected instance. In the Create Image dialog illustrated in Figure 49, you can enter a name and a brief description of your new AMI and can specify settings for its root volume. Note the Delete on Termination check box, which is marked by default. As noted in the previous section on terminating an instance, this is the default behavior but you can override it here if you want to. The tabs for EBS Volumes and Instance Storage Volumes provide a way to allocate additional storage resources to your AMI if you want to.

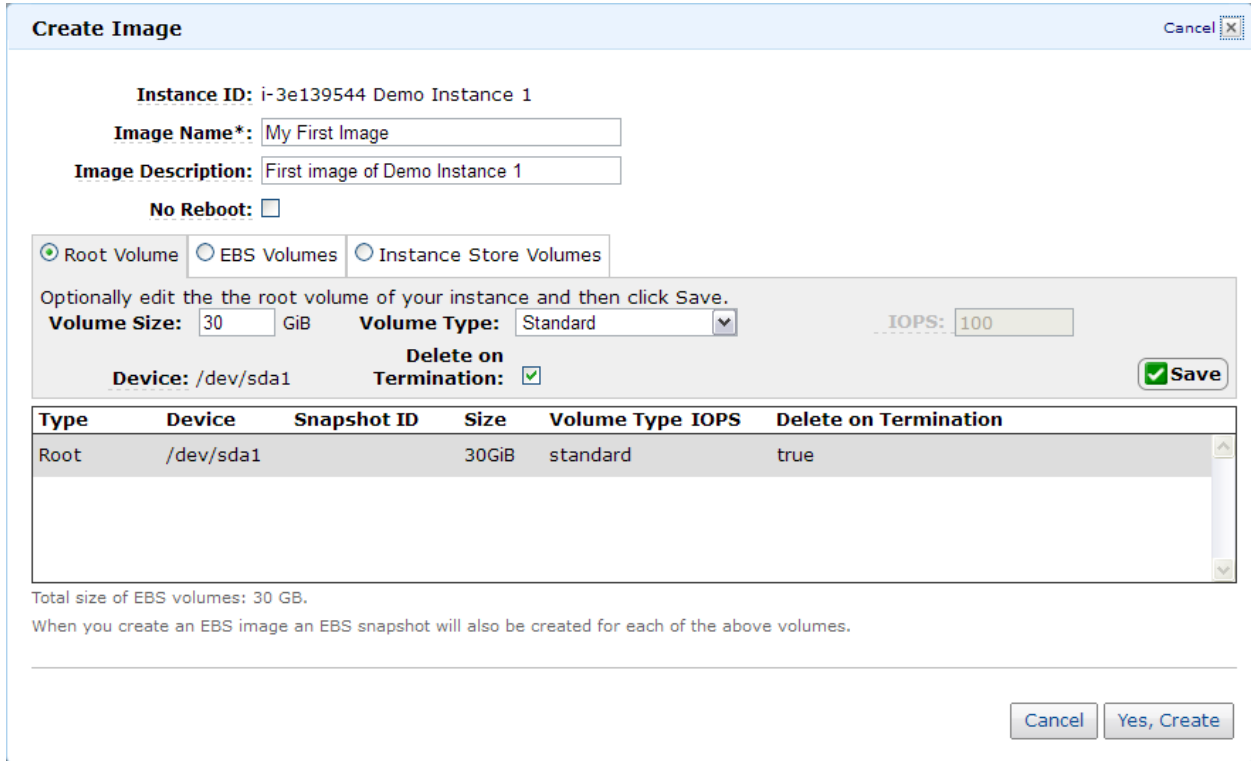


Figure 49: The Create Image dialog is where you enter the information for your new AMI.

When you're ready to proceed, click the *Yes, Create* button to begin. The process can take a while, depending on the nature of the instance and resources you're creating the AMI for. When the process is complete, the new AMI shows up in the AMIs page of the EC2 Management Console as illustrated in Figure 50.

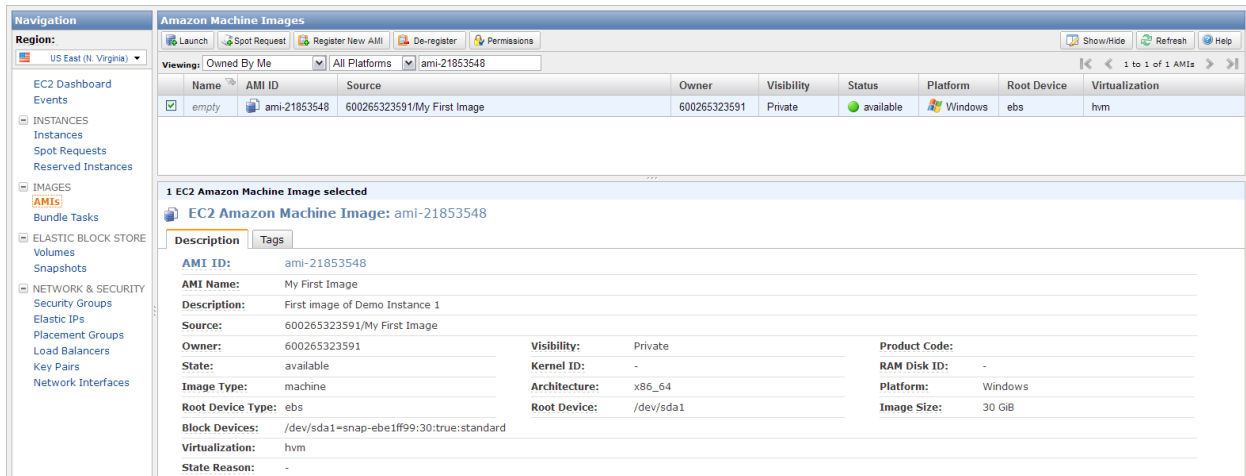


Figure 50: The AMIs page shows the newly created AMI. Note that its visibility is Private by default.

You can launch an instance from your new AMI by selecting the AMI and clicking the Launch button (upper left of the main panel in Figure 50). When the instance is running, it

shows up and can be managed like any other in the My Instances list. The new instance also gets its own EBS root volume, which is a duplicate (including any user data) of the one from which the AMI was created at the time it was created.

AMIs that you create yourself are private by default, but can be made public. One reason you might want to create a private AMI is for your own use as a quick way to launch a fully configured server instance. This can act as a hot standby in case there is a failure of the primary running instance, and can also serve as the template for launching multiple servers of the same base configuration if that's appropriate for your situation.

Once you've created your own AMI, you can make it shared and publish it so it's available to others. There is a marketplace for shared AMIs, and many are available. One way to explore them is to select the AWS Marketplace in the first step of the Create a New Instance process (see Figure 14).

Shared AMIs can be made available for free, but if you want to (or if you need to for licensing reasons) you can charge a fee for people to use them. This charge is independent of any AWS usage charges. If you publish a shared AMI and someone else launches an instance of it from a different AWS account, their account incurs the AWS usage charges, not yours.

Notes from the field

They say you can always tell the pioneer by the arrows in his back. Following are a short set of notes about things I ran into the first time I set up an AWS server, offered in the spirit of helping you avoid similar arrows during your own experiences.

Using FTP

When I set up my first AWS server I started with an AMI based on the 32-bit version of Windows Server 2008 R2 that did not include IIS. In order to configure it as a Web server, I therefore needed to install the Windows features for the IIS and FTP services, among others. Windows Server 2008 R2 shipped with IIS 7.0, but the FTP 7.0 and 7.5 services were not included. Windows therefore installed the IIS 6 version of the FTP service on my AWS server. I had successfully used this version of the Microsoft FTP server in the past on workstations, physical servers, and virtual servers hosted on services other than AWS, so I didn't give it a second thought.

However, I soon ran into trouble. While my FTP client initially appeared to connect to the FTP server using the AWS instance's elastic IP address, I found it could not get a good data connection and was therefore unable to send or receive files. The automated FTP transfer of files back and forth between the server to the client machine was an integral part of this particular application's design, so this was a show stopper.

As I was researching the problem, I found references to a limitation in the IIS 6 version of FTP that can prevent it from sending the correct IP address back to the client in order to

establish a data connection. In my past experience this had never been a problem so this was news to me, but it seemed to make sense because on AWS the server has both an public IP address and a private IP address. Evidently IIS 6 FTP was not going to work for this application.

Further research indicated that Microsoft FTP 7.5, which is available as a separate download from Microsoft, does not suffer from this same limitation. However, I had never worked with FTP 7.5 and didn't want to experiment with it on a live server just days before going into production. I chose instead to use Cerberus FTP Server because I had experience with it in the past and was confident it would work. By the way, another advantage of the Cerberus FTP server is its ability to self-generate a secure site certificate so clients can communicate via secure FTP. I don't know if the Microsoft FTP 7.5 server has this same capability.

Sending Email

Out of the box, AWS imposes limits on email sent from EC2 servers. Although I don't know for sure, it seems to me one obvious reason for this is that Amazon doesn't want people using AWS to create spam servers. In any case, if you want your applications to be able to send email from your EC2 server(s), you'll need to ask Amazon to remove the email sending limits on your account.

You can do this by simply sending an email to AWS explaining why you need the email limits removed. In my case, I explained that I was deploying an e-commerce app that needed to be able to send email confirmations to customers after they placed an order. AWS responded to my request the next day and removed the limits. This change took effect at the account level, so it is not limited to a specific server instance or a specific IP address.

Installing additional Windows features

The launch configuration you choose to start with has a bearing on the resources available to your server instance. Regardless of how much storage or computing power your server needs for its intended purpose, one place where performance really counts is when you need to install additional Windows features.

You can add Windows features to your AWS server instance the same way you would add them to a physical server. Although you might not notice it on a physical server, this is a resource-intensive process. I discovered this when installing IIS, FTP, and related Web server features to the first server I ever created. It was running as a micro instance, which comes with very limited capacities – small amount of RAM and limited I/O. As it turned out, adding these Windows features literally took hours, as in five or six hours. Clearly there had to be a better way.

One of the great things about AWS is that you can take a snapshot of an instance's hard drive, save it to persistent storage as an EBS volume, and later attach it to a different instance. This enables a solution suggested by Frank Perez, Jr. – fire up a large instance, do

all of your Windows and other configuration tasks on it, take a snapshot, terminate the large instance, fire up the micro or small instance you want to use for production, detach its default volume, and then attach your saved volume as the root device. Works like a charm.

Installing Windows updates

There's not much to say here except that it's up to you to install Windows updates on your AWS server instances. Amazon does not do this for you.

Summary

AWS comprises a vast offering of cloud-based services. This paper has focused on only one small part, namely how to launch, configure, and manage an Elastic Compute Cloud (EC2) server instance running Microsoft Windows Server. Although even this small portion of AWS is daunting enough in its complexity, it's actually fairly easy to work with once you become familiar with it. Amazon makes it possible to try out EC2 for free, so you've nothing to invest but your time. If you believe cloud-based computing is the future, it's well worth the investment.

Acknowledgements

I'd like to express my gratitude to Frank Perez, Jr. for the information and assistance he provided when I was first getting acquainted with AWS. As a new user, I was pretty overwhelmed by the scope and complexity of AWS. Frank's patient explanations helped to rapidly bring things into focus for me. I hope this paper helps you, the reader, as much as Frank helped me.

Glossary

AMI – Amazon Machine Image

Availability Zone – a location within an AWS region isolated against failures in other zones

AWS Management Console – the browser-based interface to Amazon Web Services

EBS – Elastic Block Storage; persistent storage attached to some instance types

EBS Volume – a volume of elastic block storage

EBS Snapshot – a point-in-time copy of an EBS volume

EC2 Console Dashboard – the home page for EC2 server management

EC2 Management Console – the interface for managing EC2 server instances

EC2 – Elastic Compute Cloud (aka Elastic Cloud Computing)

ECU – EC2 Compute Unit

EIP – Elastic IP Address

Instance – a virtual server derived from an AMI

Instance Hour – one instance running for one hour

Instance Storage – ephemeral storage attached to some instance types

Instance Type – a pre-packaged server configuration

Key Pair – an RSA public key / private key combination

RDP – Remote Desktop Protocol

Region – AWS has data centers in several different regions of the world

Root Volume – the EBS volume used to boot an EBS-backed server instance

S3 – Amazon Simple Storage Service (the first AWS service offered by Amazon)

Security Group – a collection of rules for controlling access to a server instance

Snapshot – see EBS Snapshot

Volume – see EBS Volume

Resources

Getting Started with Amazon EC2

<http://docs.amazonwebservices.com/AWSEC2/latest/GettingStartedGuide/Welcome.html>

Adding rules to the default security group

<http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/adding-security-group-rules.html?r=1742>

Amazon Machine Instances (AMIs)

https://aws.amazon.com/amis?_encoding=UTF8&jiveRedirect=1

AWS Documentation Home Page

<http://aws.amazon.com/documentation/>

AWS Management Console

<https://console.aws.amazon.com/console/home>

Developer tools

http://aws.amazon.com/developertools?_encoding=UTF8&jiveRedirect=1

EC2 Console Dashboard (US East Region)

<https://console.aws.amazon.com/ec2/home?region=us-east-1&#s=Home>

EC2 Home Page

<http://aws.amazon.com/ec2/>

EC2 Pricing

<http://aws.amazon.com/ec2/pricing/>

EC2 Instance Types

<http://aws.amazon.com/ec2/instance-types/>

EC2 Instance Families and Types

<http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/instance-types.html>

EC2 Purchasing Options

<http://aws.amazon.com/ec2/purchasing-options/>

EC2 Microsoft Windows Guide

<http://docs.amazonwebservices.com/AWSEC2/latest/WindowsGuide/Welcome.html>

Free tier usage

<http://docs.amazonwebservices.com/gettingstarted/latest/awsgsg-freetier/TestDriveFreeTier.html>

Monitoring Instances with Status Checks

<http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/monitoring-system-instance-status-check.html>

Copyright © 2012 Rick Borup. Windows® is a registered trademark of Microsoft Corporation in the United States and other countries. Amazon, Amazon.com, Amazon EC2, AWS, EC2, Elastic Compute Cloud, and the Amazon.com graphics and logos are trademarks or registered trademarks of Amazon. All other trademarks are the property of their respective owners.