NAME

Net::Amazon::EC2 − Perl interface to the Amazon Elastic Compute Cloud (EC2) environment.

VERSION

This document describes version 0.14 of Net::Amazon::EC2, released February 1st, 2010. This module is coded against the Query API version of the '2009−11−30' version of the EC2 API last updated December 8th, 2009.

SYNOPSIS

```
use Net::Amazon::EC2;

my $ec2 = Net::Amazon::EC2->new(
      AWSAccessKeyId => 'PUBLIC_KEY_HERE',
      SecretAccessKey => 'SECRET_KEY_HERE'
);

# Start 1 new instance from AMI: ami-XXXXXXXX
my $instance = $ec2->run_instances(ImageId => 'ami-XXXXXXXX', MinCount => 1, Max

my $running_instances = $ec2->describe_instances;

foreach my $reservation (@$running_instances) {
   foreach my $instance ($reservation->instances_set) {
      print $instance->instance_id . "\n";
   }
}

my $instance_id = $instance->instances_set->[0]->instance_id;

print "$instance_id\n";

# Terminate instance

my $result = $ec2->terminate_instances(InstanceId => $instance_id);
```

If an error occurs while communicating with EC2, the return value of these methods will be a Net::Amazon::EC2::Errors object.

DESCRIPTION

This module is a Perl interface to Amazon's Elastic Compute Cloud. It uses the Query API to communicate with Amazon's Web Services framework.

CLASS METHODS

new(%params)

This is the constructor, it will return you a Net::Amazon::EC2 object to work with. It takes these parameters:

AWSAccessKeyId (required)
: Your AWS access key.

SecretAccessKey (required)
: Your secret key, WARNING! don't give this out or someone will be able to use your account and incur charges on your behalf.

region (optional)
: The region to run the API requests through. The options are:

  - us−east−1 − Nothern Virginia
  - us−west−1 − Northern California
  - eu−west−1 − Ireland

debug (optional)
: A flag to turn on debugging. It is turned off by default

OBJECT METHODS

allocate_address()

Acquires an elastic IP address which can be associated with an instance to create a movable static IP. Takes no arguments

Returns the IP address obtained.

associate_address(%params)

Associates an elastic IP address with an instance. It takes the following arguments:

InstanceId (required)
The instance id you wish to associate the IP address with

PublicIp (required)
The IP address to associate with

Returns true if the association succeeded.

attach_volume(%params)

Attach a volume to an instance.

VolumeId (required)
The volume id you wish to attach.

InstanceId (required)
The instance id you wish to attach the volume to.

Device (required)
The device id you want the volume attached as.

Returns a Net::Amazon::EC2::Attachment object containing the resulting volume status.

authorize_security_group_ingress(%params)

This method adds permissions to a security group. It takes the following parameters:

GroupName (required)
The name of the group to add security rules to.

SourceSecurityGroupName (required when authorizing a user and group together)
Name of the group to add access for.

SourceSecurityGroupOwnerId (required when authorizing a user and group together)
Owner of the group to add access for.

IpProtocol (required when adding access for a CIDR)
IP Protocol of the rule you are adding access for (TCP, UDP, or ICMP)

FromPort (required when adding access for a CIDR)
Beginning of port range to add access for.

ToPort (required when adding access for a CIDR)
End of port range to add access for.

CidrIp (required when adding access for a CIDR)
The CIDR IP space we are adding access for.

Adding a rule can be done in two ways: adding a source group name + source group owner id, or, by Protocol + start port + end port + CIDR IP. The two are mutally exclusive.

Returns 1 if rule is added successfully.

bundle_instance(%params)

Bundles the Windows instance. This procedure is not applicable for Linux and UNIX instances.

NOTE NOTE NOTE This is not well tested as I don't run windows instances

InstanceId (required)
The ID of the instance to bundle.

Storage.S3.Bucket (required)
The bucket in which to store the AMI. You can specify a bucket that you already own or a new bucket that Amazon EC2 creates on your behalf. If you specify a bucket that belongs to someone else, Amazon EC2 returns an error.

Storage.S3.Prefix (required)
Specifies the beginning of the file name of the AMI.

Storage.S3.AWSAccessKeyId (required)
The Access Key ID of the owner of the Amazon S3 bucket.

Storage.S3.UploadPolicy (required)
An Amazon S3 upload policy that gives Amazon EC2 permission to upload items into Amazon S3 on the user's behalf.

Storage.S3.UploadPolicySignature (required)
The signature of the Base64 encoded JSON document.

JSON Parameters: (all are required)

expiration − The expiration of the policy. Amazon recommends 12 hours or longer. conditions − A list of restrictions on what can be uploaded to Amazon S3. Must contain the bucket and ACL conditions in this table. bucket − The bucket to store the AMI. acl − This must be set to ec2−bundle−read.

Returns a Net::Amazon::EC2::BundleInstanceResponse object

cancel_bundle_task(%params)
Cancels the bundle task. This procedure is not applicable for Linux and UNIX instances.

BundleId (required)
The ID of the bundle task to cancel.

Returns a Net::Amazon::EC2::BundleInstanceResponse object

confirm_product_instance(%params)
Checks to see if the product code passed in is attached to the instance id, taking the following parameter:

ProductCode (required)
The Product Code to check

InstanceId (required)
The Instance Id to check

Returns a Net::Amazon::EC2::ConfirmProductInstanceResponse object

create_image(%params)
Creates an AMI that uses an Amazon EBS root device from a ''running'' or ''stopped'' instance.

AMIs that use an Amazon EBS root device boot faster than AMIs that use instance stores. They can be up to 1 TiB in size, use storage that persists on instance failure, and can be stopped and started.

InstanceId (required)
The ID of the instance.

Name (required)
The name of the AMI that was provided during image creation.

Note that the image name has the following constraints:

3−128 alphanumeric characters, parenthesis, commas, slashes, dashes, or underscores.

Description (optional)
The description of the AMI that was provided during image creation.

NoReboot (optional)
By default this property is set to false, which means Amazon EC2 attempts to cleanly shut down the instance before image creation and reboots the instance afterwards. When set to true, Amazon EC2 does not shut down the instance before creating the image. When this option is used, file system integrity on the created image cannot be guaranteed.

Returns the ID of the AMI created.

create_key_pair(%params)
Creates a new 2048 bit key pair, taking the following parameter:

KeyName (required)
A name for this key. Should be unique.

Returns a Net::Amazon::EC2::KeyPair object

create_security_group(%params)
This method creates a new security group. It takes the following parameters:

GroupName (required)
The name of the new group to create.

GroupDescription (required)
A short description of the new group.

Returns 1 if the group creation succeeds.

create_snapshot(%params)
Create a snapshot of a volume. It takes the following arguments:

VolumeId (required)
The volume id of the volume you want to take a snapshot of.

Description (optional)
Description of the Amazon EBS snapshot.

Returns a Net::Amazon::EC2::Snapshot object of the newly created snapshot.

create_volume(%params)
Creates a volume.

Size (required)
The size in GiB of the volume you want to create.

SnapshotId (optional)
The optional snapshot id to create the volume from.

AvailabilityZone (required)
The availability zone to create the volume in.

Returns true if the releasing succeeded.

delete_key_pair(%params)
This method deletes a keypair. Takes the following parameter:

KeyName (required)
The name of the key to delete.

Returns 1 if the key was successfully deleted.

delete_security_group(%params)
This method deletes a security group. It takes the following parameter:

GroupName (required)
The name of the security group to delete.

Returns 1 if the delete succeeded.

delete_snapshot(%params)
Deletes the snapshots passed in. It takes the following arguments:

SnapshotId (required)
Either a scalar or array ref of snapshot id's can be passed in. Will delete the corresponding snapshots.

Returns true if the deleting succeeded.

delete_volume(%params)
Delete a volume.

VolumeId (required)
The volume id you wish to delete.

Returns true if the deleting succeeded.

deregister_image(%params)
This method will deregister an AMI. It takes the following parameter:

ImageId (required)
The image id of the AMI you want to deregister.

Returns 1 if the deregistering succeeded

describe_addresses(%params)
This method describes the elastic addresses currently allocated and any instances associated with them. It takes the following arguments:

PublicIp (optional)
The IP address to describe. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::DescribeAddress objects

describe_availability_zones(%params)
This method describes the availability zones currently available to choose from. It takes the following arguments:

ZoneName (optional)
The zone name to describe. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::AvailabilityZone objects

describe_bundle_tasks(%params)
Describes current bundling tasks. This procedure is not applicable for Linux and UNIX instances.

BundleId (optional)
The optional ID of the bundle task to describe.

Returns a array ref of Net::Amazon::EC2::BundleInstanceResponse objects

describe_image_attributes(%params)
This method pulls a list of attributes for the image id specified

ImageId (required)
A scalar containing the image you want to get the list of attributes for.

Attribute (required)
A scalar containing the attribute to describe.

Valid attributes are:

launchPermission − The AMIs launch permissions.
ImageId − ID of the AMI for which an attribute will be described.
productCodes − The product code attached to the AMI.
kernel − Describes the ID of the kernel associated with the AMI.
ramdisk − Describes the ID of RAM disk associated with the AMI.
blockDeviceMapping − Defines native device names to use when exposing virtual devices.
platform − Describes the operating system platform.

Returns a Net::Amazon::EC2::DescribeImageAttribute object

* NOTE: There is currently a bug in Amazon's SOAP and Query API for when you try and describe the attributes: kernel, ramdisk, blockDeviceMapping, or platform AWS returns an invalid response. No response yet from Amazon on an ETA for getting that bug fixed.

describe_images(%params)
This method pulls a list of the AMIs which can be run. The list can be modified by passing in some of the following parameters:

ImageId (optional)
Either a scalar or an array ref can be passed in, will cause just these AMIs to be 'described'

Owner (optional)
Either a scalar or an array ref can be passed in, will cause AMIs owned by the Owner's provided will be 'described'. Pass either account ids, or 'amazon' for all amazon-owned AMIs, or 'self' for your own AMIs.

ExecutableBy (optional)
Either a scalar or an array ref can be passed in, will cause AMIs executable by the account id's specified. Or 'self' for your own AMIs.

Returns an array ref of Net::Amazon::EC2::DescribeImagesResponse objects

describe_instances(%params)
This method pulls a list of the instances which are running or were just running. The list can be modified by passing in some of the following parameters:

InstanceId (optional)
Either a scalar or an array ref can be passed in, will cause just these instances to be 'described'

Returns an array ref of Net::Amazon::EC2::ReservationInfo objects

describe_instance_attribute(%params)
Returns information about an attribute of an instance. Only one attribute can be specified per call.

InstanceId (required)
The instance id we want to describe the attributes of.

Attribute (required)
The attribute we want to describe. Valid values are:

- instanceType
- kernel
- ramdisk
- userData
- disableApiTermination
- instanceInitiatedShutdownBehavior
- rootDeviceName
- blockDeviceMapping

Returns a Net::Amazon::EC2::DescribeInstanceAttributeResponse object

describe_key_pairs(%params)
This method describes the keypairs available on this account. It takes the following parameter:

KeyName (optional)
The name of the key to be described. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::DescribeKeyPairsResponse objects

describe_regions(%params)
Describes EC2 regions that are currently available to launch instances in for this account.

RegionName (optional)
The name of the region(s) to be described. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::Region objects

describe_reserved_instances(%params)
Describes Reserved Instances that you purchased.

ReservedInstancesId (optional)
The reserved instance id(s) to be described. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::ReservedInstance objects

describe_reserved_instances_offerings(%params)
Describes Reserved Instance offerings that are available for purchase. With Amazon EC2 Reserved Instances, you purchase the right to launch Amazon EC2 instances for a period of time (without getting insufficient capacity errors) and pay a lower usage rate for the actual time used.

ReservedInstancesOfferingId (optional)
ID of the Reserved Instances to describe.

InstanceType (optional)
The instance type on which the Reserved Instance can be used.

AvailabilityZone (optional)
The Availability Zone in which the Reserved Instance can be used.

ProductDescription (optional)
The Reserved Instance description.

Returns an array ref of Net::Amazon::EC2::ReservedInstanceOffering objects

describe_security_groups(%params)
This method describes the security groups available to this account. It takes the following parameter:

GroupName (optional)
The name of the security group(s) to be described. Can be either a scalar or an array ref.

Returns an array ref of Net::Amazon::EC2::SecurityGroup objects

describe_snapshot_attribute(%params)
Describes the snapshots attributes related to the snapshot in question. It takes the following arguments:

SnapshotId (optional)
Either a scalar or array ref of snapshot id's can be passed in. If this isn't passed in it will describe the attributes of all the current snapshots.

Attribute (required)
The attribute to describe, currently, the only valid attribute is createVolumePermission.

Returns a Net::Amazon::EC2::SnapshotAttribute object.

describe_snapshots(%params)
Describes the snapshots available to the user. It takes the following arguments:

SnapshotId (optional)
Either a scalar or array ref of snapshot id's can be passed in. If this isn't passed in it will describe all the current snapshots.

Owner (optional)
The owner of the snapshot.

RestorableBy (optional)
A user who can create volumes from the snapshot.

Returns an array ref of Net::Amazon::EC2::Snapshot objects.

describe_volumes(%params)
Describes the volumes currently created. It takes the following arguments:

VolumeId (optional)
Either a scalar or array ref of volume id's can be passed in. If this isn't passed in it will describe all the current volumes.

Returns an array ref of Net::Amazon::EC2::Volume objects.

detach_volume(%params)
Detach a volume from an instance.

VolumeId (required)
The volume id you wish to detach.

InstanceId (optional)
The instance id you wish to detach from.

Device (optional)
The device the volume was attached as.

Force (optional)
A boolean for if to forcibly detach the volume from the instance. WARNING: This can lead to data loss or a corrupted file system. Use this option only as a last resort to detach a volume from a failed instance. The instance will not have an opportunity to flush file system caches nor file system meta data.

Returns a Net::Amazon::EC2::Attachment object containing the resulting volume status.

disassociate_address(%params)

Disassociates an elastic IP address with an instance. It takes the following arguments:

PublicIp (required)

The IP address to disassociate

Returns true if the disassociation succeeded.

get_console_output(%params)

This method gets the output from the virtual console for an instance. It takes the following parameters:

InstanceId (required)

A scalar containing a instance id.

Returns a Net::Amazon::EC2::ConsoleOutput object.

get_password_data(%params)

Retrieves the encrypted administrator password for the instances running Windows. This procedure is not applicable for Linux and UNIX instances.

InstanceId (required)

The Instance Id for which to retrieve the password.

Returns a Net::Amazon::EC2::InstancePassword object

modify_image_attribute(%params)

This method modifies attributes of an machine image.

ImageId (required)

The AMI to modify the attributes of.

Attribute (required)

The attribute you wish to modify, right now the attributes you can modify are launchPermission and productCodes

OperationType (required for launchPermission)

The operation you wish to perform on the attribute. Right now just 'add' and 'remove' are supported.

UserId (required for launchPermission)

User Id's you wish to add/remove from the attribute.

UserGroup (required for launchPermission)

Groups you wish to add/remove from the attribute. Currently there is only one User Group available 'all' for all Amazon EC2 customers.

ProductCode (required for productCodes)

Attaches a product code to the AMI. Currently only one product code can be assigned to the AMI. Once this is set it cannot be changed or reset.

Returns 1 if the modification succeeds.

modify_instance_attribute(%params)

Modify an attribute of an instance. Only one attribute can be specified per call.

InstanceId (required)

The instance id we want to modify the attributes of.

Attribute (required)

The attribute we want to modify. Valid values are:

- instanceType
- kernel
- ramdisk
- userData
- disableApiTermination

- instanceInitiatedShutdownBehavior
- rootDeviceName
- blockDeviceMapping

Value (required)
The value to set the attribute to.

Returns 1 if the modification succeeds.

modify_snapshot_attribute(%params)
This method modifies attributes of a snapshot.

SnapshotId (required)
The snapshot id to modify the attributes of.

UserId (optional)
User Id you wish to add/remove create volume permissions for.

UserGroup (optional)
User Id you wish to add/remove create volume permissions for. To make the snapshot createable by all set the UserGroup to ‘‘all’’.

Attribute (required)
The attribute you wish to modify, right now the only attribute you can modify is ‘‘CreateVolumePermission’’

OperationType (required)
The operation you wish to perform on the attribute. Right now just ’add’ and ’remove’ are supported.

Returns 1 if the modification succeeds.

monitor_instances(%params)
Enables monitoring for a running instance. For more information, refer to the Amazon CloudWatch Developer Guide.

InstanceId (required)
The instance id(s) to monitor. Can be a scalar or an array ref

Returns an array ref of Net::Amazon::EC2::MonitoredInstance objects

purchase_reserved_instances_offering(%params)
Purchases a Reserved Instance for use with your account. With Amazon EC2 Reserved Instances, you purchase the right to launch Amazon EC2 instances for a period of time (without getting insufficient capacity errors) and pay a lower usage rate for the actual time used.

ReservedInstancesOfferingId (required)
ID of the Reserved Instances to describe. Can be either a scalar or an array ref.

InstanceCount (optional)
The number of Reserved Instances to purchase (default is 1). Can be either a scalar or an array ref.

NOTE NOTE NOTE, the array ref needs to line up with the InstanceCount if you want to pass that in, so that the right number of instances are started of the right instance offering

Returns 1 if the reservations succeeded.

reboot_instances(%params)
This method reboots an instance. It takes the following parameters:

InstanceId (required)
Instance Id of the instance you wish to reboot. Can be either a scalar or array ref of instances to reboot.

Returns 1 if the reboot succeeded.

register_image(%params)
This method registers an AMI on the EC2. It takes the following parameter:

imageLocation (optional)
The location of the AMI manifest on S3

name (required)
The name of the AMI that was provided during image creation.

description (optional)
The description of the AMI.

architecture (optional)
The architecture of the image. Either i386 or x86_64

kernelId (optional)
The ID of the kernel to select.

ramdiskId (optional)
The ID of the RAM disk to select. Some kernels require additional drivers at launch.

rootDeviceName (optional)
The root device name (e.g., /dev/sda1).

blockDeviceMapping (optional)
This needs to be a data structure like this:

```
[ {  deviceName => ‘‘/dev/sdh’’, (optional)
  virtualName => ‘‘ephermel0’’, (optional)
  noDevice => ‘‘/dev/sdl’’, (optional),
  ebs => {
    snapshotId => ‘‘snap−0000’’, (optional)
    volumeSize => ‘‘20’’, (optional)
    deleteOnTermination => ‘‘false’’, (optional) },
},  ... ]
```

Returns the image id of the new image on EC2.

release_address(%params)
Releases an allocated IP address. It takes the following arguments:

PublicIp (required)
The IP address to release

Returns true if the releasing succeeded.

reset_image_attribute(%params)
This method resets an attribute for an AMI to its default state (NOTE: product codes cannot be reset). It takes the following parameters:

ImageId (required)
The image id of the AMI you wish to reset the attributes on.

Attribute (required)
The attribute you want to reset.

Returns 1 if the attribute reset succeeds.

reset_instance_attribute(%params)
Reset an attribute of an instance. Only one attribute can be specified per call.

InstanceId (required)
The instance id we want to reset the attributes of.

Attribute (required)
The attribute we want to reset. Valid values are:

- kernel
- ramdisk

Returns 1 if the reset succeeds.

reset_snapshot_attribute(%params)

This method resets an attribute for an snapshot to its default state.

It takes the following parameters:

SnapshotId (required)
: The snapshot id of the snapshot you wish to reset the attributes on.

Attribute (required)
: The attribute you want to reset (currently ''CreateVolumePermission'' is the only valid attribute).

Returns 1 if the attribute reset succeeds.

revoke_security_group_ingress(%params)

This method revoke permissions to a security group. It takes the following parameters:

GroupName (required)
: The name of the group to revoke security rules from.

SourceSecurityGroupName (required when revoking a user and group together)
: Name of the group to revoke access from.

SourceSecurityGroupOwnerId (required when revoking a user and group together)
: Owner of the group to revoke access from.

IpProtocol (required when revoking access from a CIDR)
: IP Protocol of the rule you are revoking access from (TCP, UDP, or ICMP)

FromPort (required when revoking access from a CIDR)
: Beginning of port range to revoke access from.

ToPort (required when revoking access from a CIDR)
: End of port range to revoke access from.

CidrIp (required when revoking access from a CIDR)
: The CIDR IP space we are revoking access from.

Revoking a rule can be done in two ways: revoking a source group name + source group owner id, or, by Protocol + start port + end port + CIDR IP. The two are mutally exclusive.

Returns 1 if rule is revoked successfully.

run_instances(%params)

This method will start instance(s) of AMIs on EC2. The parameters indicate which AMI to instantiate and how many / what properties they have:

ImageId (required)
: The image id you want to start an instance of.

MinCount (required)
: The minimum number of instances to start.

MaxCount (required)
: The maximum number of instances to start.

KeyName (optional)
: The keypair name to associate this instance with. If omitted, will use your default keypair.

SecurityGroup (optional)
: An scalar or array ref. Will associate this instance with the group names passed in. If omitted, will be associated with the default security group.

AdditionalInfo (optional)
: Specifies additional information to make available to the instance(s).

UserData (optional)
: Optional data to pass into the instance being started. Needs to be base64 encoded.

InstanceType (optional)
: Specifies the type of instance to start. The options are:

m1.small (default)
1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit). 32−bit, 1.7GB RAM, 160GB disk

m1.large: Standard Large Instance
4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each). 64−bit, 7.5GB RAM, 850GB disk

m1.xlarge: Standard Extra Large Instance
8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each). 64−bit, 15GB RAM, 1690GB disk

c1.medium: High-CPU Medium Instance
5 EC2 Compute Units (2 virutal cores with 2.5 EC2 Compute Units each). 32−bit, 1.7GB RAM, 350GB disk

c1.xlarge: High-CPU Extra Large Instance
20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each). 64−bit, 7GB RAM, 1690GB disk

m2.2xlarge
13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each). 64−bit, 34.2GB RAM, 850GB disk

m2.4xlarge
26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each). 64−bit, 68.4GB RAM, 1690GB disk

Placement.AvailabilityZone (optional)
The availability zone you want to run the instance in

KernelId (optional)
The id of the kernel you want to launch the instance with

RamdiskId (optional)
The id of the ramdisk you want to launch the instance with

BlockDeviceMapping.VirtualName (optional)
This is the virtual name for a blocked device to be attached, may pass in a scalar or arrayref

BlockDeviceMapping.DeviceName (optional)
This is the device name for a block device to be attached, may pass in a scalar or arrayref

Encoding (optional)
The encoding.

Version (optional)
The version.

Monitoring.Enabled (optional)
Enables monitoring for this instance.

SubnetId (optional)
Specifies the subnet ID within which to launch the instance(s) for Amazon Virtual Private Cloud.

Returns a Net::Amazon::EC2::ReservationInfo object

start_instances(%params)
Starts an instance that uses an Amazon EBS volume as its root device.

InstanceId (required)
Either a scalar or an array ref can be passed in (containing instance ids to be started).

Returns an array ref of Net::Amazon::EC2::InstanceStateChange objects.

stop_instances(%params)
Stops an instance that uses an Amazon EBS volume as its root device.

InstanceId (required)
Either a scalar or an array ref can be passed in (containing instance ids to be stopped).

Force (optional)
If set to true, forces the instance to stop. The instance will not have an opportunity to flush file system caches nor file system meta data. If you use this option, you must perform file system check and repair procedures. This option is not recommended for Windows instances.

The default is false.

Returns an array ref of Net::Amazon::EC2::InstanceStateChange objects.

terminate_instances(%params)
This method shuts down instance(s) passed into it. It takes the following parameter:

InstanceId (required)
Either a scalar or an array ref can be passed in (containing instance ids)

Returns an array ref of Net::Amazon::EC2::InstanceStateChange objects.

unmonitor_instances(%params)
Disables monitoring for a running instance. For more information, refer to the Amazon CloudWatch Developer Guide.

InstanceId (required)
The instance id(s) to monitor. Can be a scalar or an array ref

Returns an array ref of Net::Amazon::EC2::MonitoredInstance objects

TESTING

Set AWS_ACCESS_KEY_ID and SECRET_ACCESS_KEY environment variables to run the live tests. Note: because the live tests start an instance (and kill it) in both the tests and backwards compat tests there will be 2 hours of machine instance usage charges (since there are 2 instances started) which as of February 1st, 2010 costs a total of $0.17 USD

Important note about the windows-only methods. These have not been well tested as I do not run windows-based instances, so exercise caution in using these.

TODO

Need to add in support for Spot Instances.

AUTHOR

Jeff Kim <cpan AT chosec DOT com>

CONTRIBUTORS

John McCullough

COPYRIGHT

SEE ALSO

Amazon EC2 API: <http://docs.amazonwebservices.com/AWSEC2/latest/APIReference/>